Artificial Chemistries - A Review

Peter Dittrich, Jens Ziegler, and Wolfgang Banzhaf

University of Dortmund
Department of Computer Science
Systems Analysis
D-44221 Dortmund
Germany

http://ls11-www.cs.uni-dortmund.de

dittrich,ziegler,banzhaf@ls11.cs.uni-dortmund.de

June 13, 2001

Abstract

This article reviews the growing body of scientific work in artificial chemistry. First, common motivations and fundamental concepts are introduced. Second, current research activities are discussed along three application dimensions: modeling, information processing and optimization. Finally, common phenomena among the different systems are summarized. It is argued here that artificial chemistries are "the right stuff" for the study of pre-biotic and bio-chemical evolution, and they provide a productive framework for questions regarding the origin and evolution of organizations in general. Furthermore, artificial chemistries have a broad application range of practical problems as shown in this review.

Contents

Information should be generated from information just as organisms from organisms. The pieces should fertilize each other, they should be crossed over, they should be mutated, that is varied to a small degree, but also to a larger degree by radical changes not known in genetics. This could perhaps happen in some vessels where reactions between "information-carrying molecules" take place, molecules who carry information in a similar way as chromosomes carry the features of organisms[1]

Stanislaw Lem, Summa Technologiae, 1964

1 Artificial Life and Artificial Chemistry

One of the main driving forces of science is the quest for understanding the origin and nature of life. Since centuries, this question has caused scientists to collect and systematically describe the diversity of life found everywhere around us. Over many years an impressive collection of data about the processes of life has been amassed. Biology as the traditional Life science has sorted and classified the data, and - in the course of a century - has made considerable inroads into the understanding of certain aspects of life. The informational perspective, however, has been not at the center of interest in biological attempts to approach the issue of life's essence. *Artificial Life* (AL) research, on the other hand, abstracts from specific examples of living processes and tries to integrate different approaches into one interdisciplinary attempt to extract the first principles of life. The working hypothesis of AL research is that biotic phenomena can be modelled by using complex systems of many interacting components. The complex system approach towards an explanation of life employs *emergence* as a central concept. Emergence is used to deduce global properties of a system from the local interactions between its subsystems. These local interactions may follow simple effective rules which cause global behavior of the system to emerge, but cannot be predicted by simply analyzing the subsystems and their components. In other words, a system has certain properties not due to properties of its constituents but due to their *organization* and their mutual *function* in the whole.

If we tentatively accept the hypothesis that properties of a system's components are not the main part of the description of that system's organization, we come to the conclusion that natural systems such as organisms or social structures, though consisting of truly different matter and components might follow the same organizational principles. This fundamental idea characterizes many AL approaches. Under this view living organisms are alive not because of the properties of their constituents but of their organization.

The theory of evolution, formulated by Darwin, has caused a storm of controversy, challenging both scientific and popular beliefs about the appearance of life on earth. Up to this day the Darwinian revolution has not yet been realized in its extent nor is it accepted by most of our contemporaries. The principle of random variation and

[1] Translated by the authors from the German edition, 1976, Insel Verlag: Frankfurt a. M.

competitive selection that Darwin discovered is a powerful means for understanding the progress of evolution. It seems to be valid also on the level of replicating molecules and can be observed in higher level systems (even social or cultural systems).

What has been left open, however, by the theory of evolution are questions relating to the origin of evolutionary units: How come the prerequisites into being, i.e. the entities that are varied and selected? How did qualitatively new evolutionary mechanisms emerge, such as sexual recombination, regulation of mutation rates, or the genetic code?

Abstract models that should be able to explain the origin of evolutionary systems would allow to investigate the theoretical conditions for the origin and evolution of life. This is one of the fundamental goals of the AL subfield of *artificial chemistry* (AC) research. By abstracting from natural molecular processes, AC tries to investigate the dynamics of these complex systems. AC deals with combinatorial elements which change or maintain themselves and especially with systems that are able to construct new components. AC thus deals with forms of organization, self-maintenance, self-construction and with the conditions for those structures to arise. We would like to argue that artificial chemistries are "the right stuff" to study when trying to uncover the basic mechanisms of life, and more general even, the origin and evolution of organisations.

The spectrum of AC research is broad. Its application may be considered along three main-dimensions: (1) modeling, (2) information processing, and (3) optimization. Along the axis of *modeling*, there are several examples of Artificial Chemistry modeling systems in different domains. They range from the above mentioned biological or evolutionary systems to social systems or models of parallel processing. The metaphor of colliding molecules is their common relation to chemistry.

In the area of *information processing* the computational properties of chemical systems are investigated and exploited. Many instances of chemical processes in nature can be interpreted as performing computations. For example, chemical reaction networks control the movement of bacteria, other chemical processes control the growth of neurons during the development of a brain, and the immune system can also be regarded as a chemical information processing system. The area of information processing can be subdivided again into two parts: (a) real chemical computing and (b) artificial chemical computing. The former deals with real chemistry and tries to use real molecules in order to compute. The latter applies the chemical metaphor as a design paradigm for new hardware and software architectures.

In the area of *optimization* artificial chemical systems help to find solutions of "difficult", mostly combinatorial problems. This application domain is closely related to the field of evolutionary computing because many evolutionary algorithms can be seen as artificial chemical systems.

This review is organized as follows: First it outlines the common motivation for formulating ACs. Then it gives a structured overview of different approaches by means of common and distinctive features (Sec. 2.1). The following section (Sec. 2.2) gives a short tutorial on ACs, introducing two illustrative examples. Section 2.3 is devoted to a systematic attempt of classifying the different AC approaches. There a basis of discriminative features is given, according to which the consecutive lineup in Chapter 3 is structured. The next chapter (Ch. 4) gives an overview over projects that make use of some of the properties of AC in applications. Chapter 5 deals with observed phenomena common to

several approaches and tries to summarize their results. The closing part (Ch. 6) gives an outlook of future directions and challenges in the area.

2 Basic Concepts

This chapter gives an introduction to the basic concepts of artificial chemistries. Then follow two descriptive examples which should demonstrate concrete implementations. Finally, we try to structure the characteristics of ACs.

2.1 What is an Artificial Chemistry?

Let us start with a broad definition: *An artificial chemistry is a man-made system which is similar to a real chemical system.* This definition has been kept as general as possible in order not to exclude any relevant work. When we now become more precise in describing the structure of an artificial chemistry we should keep in mind that not all AC approaches can be subsumed under the following conceptual framework.

More formally, an *artificial chemistry* can be defined by a triple (S, R, A), where S is the set of all possible molecules, R is a set of collision rules representing the interaction among the molecules, and A is an algorithm describing the reaction vessel or domain and how the rules are applied to the molecules inside the vessel. Both, the set of molecules S and the set of reaction rules R, can be defined explicitly or implicitly (e.g., by an algorithm or mathematical expression). This will be illustrated by two examples in Sec. 2.2 and discussed in more detail in Sec. 2.3.

2.1.1 The Set of Molecules S

The set of molecules $S = \{s_1, \ldots, s_i, \ldots, s_n\}$, where n might be infinite, describes all valid molecules which may appear in an AC. A vast variety of molecule definitions can be found in different approaches. For example, molecules may be abstract symbols [47], character sequences [9, 48, 74, 92], lambda-expressions [49], binary strings [13, 40, 129], numbers [11], hierarchical tree data structures [21, 22, 96], combinators [117], or proofs [52]. A molecule's representation is often referred to as its *structure* and is set in contrast to its *function* which is given by the reaction rules R. The description of valid molecules and their structure is usually the first step in the definition of an AC. This is analogous to a part of chemistry which describes what kind of atomic configurations form stable molecules and how these molecules appear.

2.1.2 The Set of Rules R

The set of reaction rules R describes the interactions between molecules $s_i \in S$. A rule $r \in R$ can be written according to the chemical notation of reaction rules in the form

$$s_1 + s_2 + \cdots + s_n \rightarrow s'_1 + s'_2 + \cdots + s'_m. \tag{1}$$

A reaction rule determines the n components (objects, molecules) on the left hand side that can react and subsequently be replaced by the m components on the right hand side. n may be called the *order* of the reaction[2]. Note that the sign "+" is not an operator, here, but should only separate the components on either side.

[2]Note that according to chemistry this is a simplification.

A rule is applicable only if certain conditions are fulfilled. The major condition is that all of the left hand side components must be available. This condition can be broadened easily to include other parameters such as neighborhood, rate constants, probability for a reaction, or energy consumption. In such a case a reaction rule would also contain additional information or further parameters. Whether or not these additional predicates are taken into consideration depends on the objectives of the AC. If it is meant to simulate real chemistry as accurate as possible, it is necessary to integrate these parameters into the simulation system. If the goal is to build an abstract model, many of these parameters can be omitted.

2.1.3 Reactor Algorithm A – Dynamics

An algorithm determines how the set R of rules is applied to a collection of molecules P, called *reactor*, *soup*, *reaction vessel* or *population* [3]. Note that P cannot be identical to S since some molecules might be present in many exemplars, others not at all.

Algorithm A depends on the representation of P. In the simplest case, without a spatial structure in P, the population can be represented explicitly as a multi-set or implicitly as a concentration vector.

We shall now summarize how the dynamics of a reaction vessel (which contains usually a huge number of molecules) can be modeled and simulated. The approaches can be characterized roughly by whether each molecule is treated explicitly or whether all molecules of one type are represented by a number, their frequency or concentration.

(1) Stochastic molecular collisions: In this approach every molecule is explicitly simulated and the population is represented as a multi-set P. A typical algorithm draws a sample of molecules randomly from the population P and checks whether a rule $r \in R$ can be applied. If so, the molecules are replaced by the right hand side molecules given by r. If more than one rule can apply, a decision is implemented which rule to employ. If no rule can be applied, a new random drawing is initialized. The algorithm is, however, not necessarily restricted to be such simple. Further parameters such as rate constants, energy, spatial information, or temperature can be introduced into the rules for the chemistry to become more realistic.

The following example is an algorithm used for an AC with second-order reactions only:

[3] Here, the term *population* is used as a technical term according to its meaning in the field of evolutionary computation and artificial life. It refers to a data structure which holds all individuals during a simulation. It should not be confused with the technical term "population" used in biology which refers to a group of similar, interbreeding organisms that live in a particular area.

while $\neg terminate()$ **do**
  $s_1 := draw(P)$;
  $s_2 := draw(P)$;
  if $\exists\,(s_1 + s_2 \rightarrow s'_1 + s'_2 + \cdots + s'_m) \in R$
    then
      $P := remove(P, s_1, s_2)$;
      $P := insert(P, s'_1, s'_2, \ldots, s'_m)$;
  fi
od

The function *draw* returns a randomly chosen molecule from P (without removing it from P). The probability that a specific type is returned is proportional to the concentration of this type in P. The above algorithm does not include an influx or dilution flux of molecules. This can, however, be added easily.

Memory consumption of the algorithm is generally of order $O(M)$ (that is, the memory needed to store all molecules is linearly dependent on M, the total number of molecules in the reaction vessel). The explicit simulation of every collision is the lowest level of description. It is very realistic and circumvents some difficulties generated by the collective description we discuss below. There are, however, certain disadvantages for an explicit simulation of every collision. If rate constants or concentrations of molecular species differ by several orders of magnitude the atomic simulation is not efficient. In addition, if the total number of different molecular species is low or the population is large, an explicit simulation will be slow (see Sec. 2.1.4).

(2) Continuous differential or discrete difference equations: A common approach to describe the dynamics of a chemical system is to use differential rate equations which reflect the development of the concentrations of molecular species. The index carried by a molecule now characterizes its species, not its place in the population. A reaction r can be written as

$$r : a_1 s_1 + a_2 s_2 + \cdots + a_N s_N \rightarrow b_1 s_1 + b_2 s_2 + \cdots + b_N s_N \; .$$

where the coefficients a_i, b_i are the stoichiometric factors of the reaction. They are zero if a species s_i does not participate in the reaction ($a_i = 0$ if s_i is not a reactant and $b_i = 0$ if it is not a product of the reaction). Let N be the total number of different species in S. The change of the overall concentration[4] of s_i is expressed with the following system of differential equations Begin

$$\frac{\mathrm{d}s_i}{\mathrm{d}t} = (b_i - a_i) \prod_{j=1}^{N} s_j^{a_j} \quad , i = 1, \ldots, N. \tag{2}$$

In order to take into account every reaction $r \in R$ that possibly contributes to changes of concentration s_i one summarizes End Begin

[4]Note that s_i is also used to describe the concentration of the species s_i which is normally written as $[s_i]$

$$\frac{\mathrm{d}s_i}{\mathrm{d}t} = \sum_{r \in R} \left[(b_i^r - a_i^r) \prod_{j=1}^{N} s_j^{a_j^r} \right] \quad , i = 1, \ldots, N. \tag{3}$$

End

Rate equations are a continuous model for a discrete situation. The modeling and simulation of the dynamics by differential equations is an approximation for large numbers of molecules from the same species being present.

Disadvantages of this approach are: Due to the properties of the numerical solution, concentrations are floating point values and may be so small as to be below the threshold which indicates a single molecule in the reactor. For example, if the assumed population size is 1000, then every further calculation of reactions with s_i participating causes unnecessary computational effort if $[s_i]$ is below 0.001 because in fact the species s_i is no longer present in the reactor. Thus, for the ODE approach to be useful, the number of different molecular species N should be small.

(3) Metadynamics: This approach towards describing the reaction system acknowledges that the number of species and therefore the number of differential equations may change over time [9]. The equations at a given time represent the dominant species (a dominant species is a species with a concentration above a certain threshold). As the concentrations change the dominant species may change, too. Thus the differential equation system is modified by adding and/or removing equations. Bagley et al. distinguish between deterministic and stochastic metadynamics. Deterministic metadynamics: The reaction graph changes relatively to a concentration threshold ([10], [p. 144]). The sequence of graphs is purely deterministic. It explores only the internally catalyzed pathways. On the other hand, there is stochastic metadynamics: The physical accuracy of (2) is combined with the speed of the deterministic metadynamics approach.

(4) Mixed approaches: There are also approaches where single macro-molecules are simulated explicitly and a small number of molecules are represented by their concentrations [137, 136].

(5) Symbolic analysis of the equations: If the differential equation system (3) is simple enough a symbolic analysis is possible. By solving the equations symbolically the steady state behavior of the system (fixed point, limit cycle, chaotic behavior, ...) can be derived. In the mathematical sense, the symbolic approach is the most exact one. It can be combined with the metadynamics approach to calculate the dynamical fixed point of the differential equations "in one step". [9, 10, 48] used this method with their polymer reaction network which resulted in a special ODE with only one stable fixed point. In the general case, the system is neither restricted to stable attractors nor to fixed points. Dynamical behavior of the system, however, cannot be derived from the symbolic analysis because it gives insight into the asymptotic behavior only.

2.1.4 A Comparison of the Computational Costs

To ease the decision about what kind of approach to favor for the dynamic simulation in a particular situation we will now compare roughly the computational costs of using

explicit molecular collisions vs. an ODE system. For the comparison we assume that E is the mean value of the number of molecules participating in each reaction r in R, and N is the number of different species in a population P of size M. We compare the computational effort necessary to perform a single step of the two major approaches, the explicit simulation of reactions and the continuous differential equation approach.

Explicit stochastic molecular collision: In an explicitly performed simulation, a reaction requires $O(E)$ *draw*-operations, followed by $O(R)$ lookups in the reaction rule set R. Thereafter $O(E)$ *insert*-operations are performed for an intermediate update. This leads to the estimation of $O(E)+O(R)$ costs for each reaction, which equals $O(R)$, because in almost every chemistry the total number of different reactions is much larger than the average number of reactands of a single reaction. A full update of the population P thus requires a time of $O(R) \cdot O(M) = O(R \cdot M)$. These computational costs reduce to $O(M)$ if the lookup of a reaction can be done in $O(1)$ which is, e.g., the case if the reaction is implicitly defined in the structure of a molecule. However, the comparison of the costs in table (1) assumes an $O(R)$ lookup.

Numerical integration of the ODE equations (Eq. 3): An integration of the system requires N differential equations, each of which is of order E. The integration has the step length h. An update of the concentration vector $\vec{s}$ with a numerical integration method thus has computational costs of $O(\frac{1}{h} \cdot N \cdot E)$ function evaluations, because $\frac{1}{h}$ evaluations of the system with N equations of order E are needed to perform a unit step. So $O(\frac{1}{h} \cdot N \cdot E)$ steps have to be performed to have the same progress as M reactions would generate in an explicit simulation with a population P of size M. If a very common integration method like the standard Runge-Kutta method is used, an additional factor of 4 (because of the intermediate steps) needs to be weighed in, so that the overall costs sum up to $O(4 \cdot \frac{1}{h} \cdot N \cdot E)$, which equals $O(\frac{1}{h} \cdot N)$. Note that the Runga-Kutta method uses a fixed step length. If the solver uses a variable step length, the parameter h changes accordingly (variable step length solvers may be necessary if the rate constants of the reactions differ by several orders of magnitude which in turn makes the system become stiff).

Though constant factors were omitted in the assumption of runtime-behavior expressed in $O(\dots)$ terminology, they may play an important role in the calculation of the actual computational effort.[5] Results of the comparison of the two approaches are shown in Table 1. Integration is advantageous if the population is large and the number of species is small. In the opposite case the explicit simulation shows better efficiency. Which approach to favor if both, population size and number of species are large, is an unresolved problem and will depend on additional parameters.

2.1.5 Alternative Definition of an Artificial Chemistry

Begin

There is an alternative way to define an artificial chemistry more naturally, namely by a tuple (S, I) where S is a set of particles (basic building blocks, functional groups, or

[5]This comparison does not consider other effects such as creating the reaction table, or the creation and deletion of new species during the evolution of the system. It also neglects the memory requirements of both approaches which cause additional computational effort. Nevertheless, this comparison gives as a result a coarse evaluation of the approaches.

Parameters			Costs	
Popsize M	Species N	Reactions R	Collisions	Integration
10^3	10	10	10^4	10^3
10^3	10	10^2	10^5	10^3
10^3	10	10^4	10^7	10^3
10^3	10^4	10	10^4	10^6
10^3	10^4	10^2	10^5	10^6
10^3	10^4	10^4	10^7	10^6
10^3	10^8	10	10^4	10^{10}
10^3	10^8	10^2	10^5	10^{10}
10^3	10^8	10^4	10^7	10^{10}
10^6	10	10	10^7	10^3
10^6	10	10^2	10^8	10^3
10^6	10	10^4	10^{10}	10^3
10^6	10^4	10	10^7	10^6
10^6	10^4	10^2	10^8	10^6
10^6	10^4	10^4	10^{10}	10^6
10^6	10^8	10	10^7	10^{10}
10^6	10^8	10^2	10^8	10^{10}
10^6	10^8	10^4	10^{10}	10^{10}
10^9	10	10	10^{10}	10^3
10^9	10	10^2	10^{11}	10^3
10^9	10	10^4	10^{13}	10^3
10^9	10^4	10	10^{10}	10^6
10^9	10^4	10^2	10^{11}	10^6
10^9	10^4	10^4	10^{13}	10^6
10^9	10^8	10	10^{10}	10^{10}
10^9	10^8	10^2	10^{11}	10^{10}
10^9	10^8	10^4	10^{13}	10^{10}

Table 1: *Comparison of the computational costs (in instructions). Fixed step size* $h = 0.01$, E *= 2 (average reaction order). See text for more details.*

atoms) and I a description of the interactions among particles. The advantage of this definition is that the "artificial" separation into reaction rules (which specify the "logic" of the transformation of molecules) and algorithm (which defines the dynamics) is avoided. The form (S, I) is preferable if interactions or reactions are taking place in the space of the particles or molecules, like in a lattice molecular system (Sec. 3.7). In most cases the reaction between molecules is independent from the description of the reaction vessel and their movement in that vessel, like in Fontana's λ chemistry (Sec. 3.1.4). Under those conditions the form (S, R, A) is preferable.

The difficulty for separating reactions from dynamics becomes especially clear in Sec. 3.6 (assembler automata) and Sec. 3.8.1 (mechanical artificial chemistries). But in order to be consistent we use the more general form (S, R, A) throughout this paper. End

2.2 Two Examples

The first example demonstrates the relation of an explicit simulation of molecules to their differential equation model. The second example is a constructive AC where molecules and reactions are defined implicitly. The algorithms are given in detail below.

2.2.1 A Non-Constructive Explicit Chemistry

In the following example, we define a simple chemical system which consists of two molecular types. There are only deterministic second-order catalytic reactions allowed. Thus a collision of two molecules will catalyze the formation of a specific third molecule. The two colliding molecules are regarded as catalysts so that they are not changed during the collision. In order to prevent an unlimited growth of the population we assume a maximum reactor size of M and a dilution flux which keeps the number of molecules M constant in the reactor.

Molecules: Let the set of molecules be $S = \{A, B\}$. Note that A and B are symbols representing the molecules and not variables or patterns.

Reactions: The reaction rules are given explicitly as follows:

$$\begin{aligned} r_1: &\quad A + A \longrightarrow A + A + B \\ r_2: &\quad A + B \longrightarrow A + B + B \\ r_3: &\quad B + A \longrightarrow B + A + B \\ r_4: &\quad B + B \longrightarrow B + B + A \end{aligned}$$

It may also be convenient to represent the reaction rules by a *reaction table*:

	reactands		products	
rules	A	B	A	B
r_1	2		2	1
r_2	1	1	1	2
r_3	1	1	1	2
r_4		2	1	2

or, shortly

	A	B
A	B	B
B	B	A

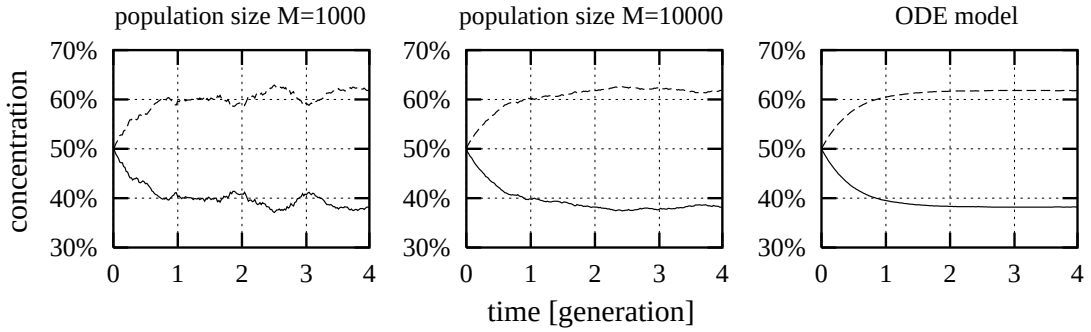


Figure 1: *Comparison of the accuracy between the explicit simulation of collisions and the numerical integration. The upper curve represents the concentration of molecular species B. Left: Simulation of a population with* 10^3 *elements. Middle: Simulation of a population with* 10^4 *elements. Right: Numerical integration of the ODE-Model*

Dynamics by explicit stochastic reactions: A simulation in which every molecule is explicitly stored and every single collision is explicitly performed can easily be implemented if the population is represented by an array P of fixed size M. An array element $P[i] \in S$ represents a single molecule. The algorithm becomes:

while $\neg terminate()$ **do**
$\quad P[randInt(1, M)] := reaction(P[randInt(1, M)], P[randInt(1, M)]);$
od

One iteration of the while-loop simulates one collision of two molecules. The algorithm is so short here because every collision results in a reaction and the production of exactly one new molecule which replaces one randomly chosen molecule. The replaced molecules form the dilution flux.

Dynamics by an ODE model: To simulate the reaction system using an ODE model we represent the population by a concentration vector $\vec{x} = (x_A, x_B)$ where x_A and x_B denote the concentration of molecular types A and B, respectively. Because the total number of molecules is kept constant we can normalize x_A and x_B such that $x_A + x_B = 1, x_A \geq 0, x_B \geq 0$ holds. The corresponding differential equations read:

$$\begin{aligned} \frac{dx_A}{dt} &= x_B x_B - x_A \Phi(t) \\ \frac{dx_B}{dt} &= x_A x_A + 2 x_A x_B - x_B \Phi(t) \qquad \text{with} \quad \Phi(t) = (x_A + x_B)^2 = 1 \end{aligned} \tag{4}$$

The dilution flux $\Phi(t)$ equals one because as a result of a collision exactly one molecule is created so that one molecule is removed at each time step.

Comparison: In Fig. 1 a comparison of the explicit stochastic collision and the ODE approach is shown. For the explicit simulation the typical behavior for two population sizes is depicted. It is clear that the smaller the population the more influence will be carried by the randomness of collisions. The ODE-Model can be seen as simulation with infinite population size and thus generates smooth concentration curves.

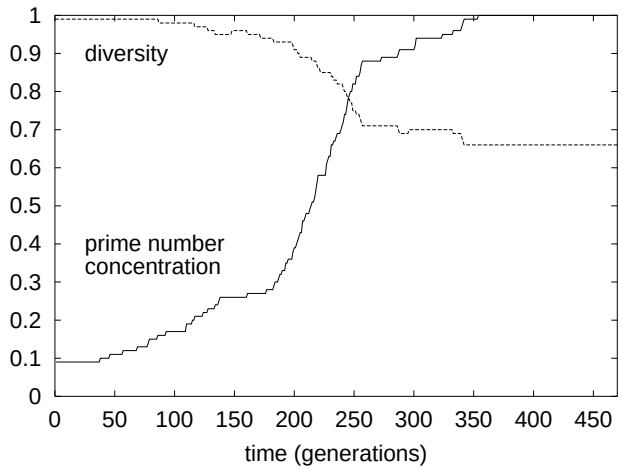


Figure 2: *Development of prime number concentration and diversity. Population size* $M = 100$*. Population initialized with number taken randomly from* $S = \{2, 3, \ldots, 10000\}$

2.2.2 A Constructive Implicit Chemistry

As example for an implicitly defined artificial chemistry we choose the **number-division chemistry** [20, 11].

Molecules: The set of molecules S are all natural numbers greater than one: $S = \{2, 3, 4, \ldots\}$.

Reactions: Mathematical division is used to calculate the reaction product. When two molecules $s_1, s_2 \in S$ collide and s_1 can divide s_2 without remainder, s_2 is transformed to s_2/s_1. Thus s_1 acts as a catalyst. The set of reactions can also be written implicitly as

$$R = \{s_1 + s_2 \rightarrow s_1 + s_3 : s_1, s_2, s_3 \in S \wedge s_3 = s_1/s_2\} \tag{5}$$

If two molecules collide and there is no corresponding reaction rule in R, the collision is elastic.

Dynamics: The algorithms draws at random two molecules out of the population and replaces one of them if it can be divided by the other reactand. So, the algorithm is similar to previous one (Sec. 2.2.1), but additionally includes elastic collisions.

Discussion: Despite of the simple algorithm, the dynamic behavior of the system is fairly interesting. After repeated application of the reaction rule, the whole population tends to eliminate non-prime numbers until it finally consists of primes only (Fig. 2)[6].

The concentration of primes shows a typical growth and finally reaches a maximum at $c_{primes} = 1.0$. The diversity reduces from its initial value to a smaller one through a transient maximum. The AC emerges as a prime factor calculation of the numbers in the reactor. By dividing larger numbers into smaller numbers, the total number of different numbers increase, until all numbers are divided into their prime factors. The concentration of prime factors increases and thus the diversity of the population decreases. Finally, all numbers in the reactor are prime factors of the initial population.

[6]Simulation in Java available at: http://ls11-www.informatik.uni-dortmund.de/alife/primegen/

2.3 Characteristics and Methods

Reihenfolge der folgenden Abschnitte geändert!

In this section, basic terms are introduced and frequently used methods are summarized. Both allow to characterize and classify artificial chemistries.

2.3.1 Definition of Molecules: Explicit or Implicit

In an artificial chemistry $\{S, R, A\}$ the set S represents the objects or molecules which interact according to certain rules defined in R. The set S can be defined explicitly or implicitly.

Molecules are *explicitly* defined if the set S is given as an enumeration of symbols, e.g., $S = \{A, B, C\}$. An *implicit* definition is a description of how to construct a molecule. This description may be a grammar. Examples for implicit definitions are: $S = \{0, 1\}^*$, the set of all bit strings; or $S = \{1, 2, 3, \dots\}$, the set of natural numbers.

To build constructive dynamical systems (Sec. 2.3.5) it is convenient to define molecules implicitly. Typical implicitly defined molecules are character sequences (e.g., *abbaab*), mathematical objects (e.g., numbers), or compound objects which consist of different elements. Compound objects can be represented by data structures [90]. We will refer to the representation of an molecule as its *structure*.

In some artificial chemistries, the structure of a molecule is not a priori defined. Instead, the arrival of molecules is an *emergent* phenomenon and interpreting a structure as a molecule is possible only a posteriori. Hence, it is not even necessary to define a molecule, neither explicitly nor implicitly.

2.3.2 Definition of Reaction Laws: Explicit or Implicit

Once we have defined the molecules of our AC, it is time to specify their interactions. These interactions can be defined, again, in two different ways.

Explicit: The definition of the interaction between molecules is independent of the molecular structure. Molecules are represented by abstract interchangeable symbols and the total number of possible elements of the AC remains fixed. Reaction rules are enumerated and explicitly given. All possible elements and their behavior is known a priori and their interaction rules do not change during the experiments.

Implicit: The definition of the interaction between molecules must refer to their structure. Examples for implicit reaction rules are concatenation or cleavage of polymers. An artificial chemistry with an implicit reaction scheme allows to derive the outcome of a collision from the structure of the colliding molecules. The number of possible molecules can be infinite, because there is no need for an explicitly defined interaction scheme. Implicitly defined interactions are commonly used for constructive chemistries.

2.3.3 Definition of Dynamics: Explicit or Implicit

Begin

The algorithm A of the artificial chemistry $\{S, R, A\}$ now needs to be defined. The *explicit* definition of the system's dynamics has already been given in Sec. 2.1.3. This definition is necessary from a computational point of view, because in a computer only the effective execution of a series of formally defined interactions causes the system to

develop and to show a dynamic behavior. The explicit definition of the dynamics is based on the interactions determined through R and may include various parameters, ranging from temperature, pressure, pH-value to field effects of secondary or higher level molecule structures, resulting in an artificial chemistry that can be used in the field of Computational Chemistry [32]. It is shown in Sec. 2.2, however, that even a random execution of interactions with no additional parameters causes interesting dynamical phenomena.

Sometimes, the definition of interactions makes the additional definition of A obsolete. This *implicit* definition is used, for example, in a cellular automata chemistry, where the dynamics is caused by the synchronous or asynchronous update of lattice sites. End

2.3.4 Levels of Abstraction: Analogous or Abstract

Artificial chemistries may also be characterized according to their level of abstraction. If there is an isomorphism between a molecule or reaction of the AC to a molecule or reaction in Chemistry, respectively. the AC can be called *analogous* otherwise *abstract*. Analogous ACs are considered in the realms of Computational Chemistry where the goal is to model chemical processes in the computer as closely as possible. What makes an abstract AC a model of Chemistry, however, are merely statistical or qualitative features of the reaction laws. The level of abstraction is varying and depends on the goals of the modeling.

2.3.5 Constructive Dynamical Systems

In a *constructive dynamical system* new components can appear which may change the dynamics of the system. This is different from a conventional dynamical system where all components and interactions are given explicitly at the outset of the process (see first example, above).

If new components are generated randomly a constructive system is called *weakly constructive* [53]. It is called *strongly constructive* if new components are generated through the action of other components. Natural Chemistry is considered to be a strongly constructive system [49].

Modelling the dynamical system with differential equations, one can describe a constructive system as an ordinary differential equation (ODE) system where equations are modified, added and/or removed dynamically. A constructive dynamical system may also be modeled by a fixed ODE system with an infinite state space. Mathematical techniques are available to handle such systems, but this approach is not intuitive and has many additional drawbacks. Using explicitly simulated molecules the system can be considered constructive if the number of elements that appeared in the system at least once increases in time.

2.3.6 Random Chemistries

One of the easiest ways to generate an artificial chemistry is by drawing random numbers for rate constants or by generating explicit reaction laws randomly. The random autocatalytic systems investigated in [118], the GARD system [114], or the metabolic networks in the multicellular organisms reported in [54] are examples for non-constructive random

chemistries. It is also possible to generate constructive chemistries randomly by drawing parameters for the reaction laws "on the fly" when they are needed as, e.g., demonstrated by Bagley and Farmer [9]. The advantage of random chemistries is that the statistical characteristics of the reaction mechanism can be specified arbitrarily. There is, however, an increasing memory consumption through newly apprearing molecular species, because every randomly generated reaction law has to be stored explicitly for the whole simulation[7].

2.3.7 Measuring Time

One step of the algorithm (10) can be interpreted as a collision of two (or more) molecules. Simulated time is proportional to the number of collisions divided by reactor size M. It is common to measure the simulated time in generations, where one *generation* consists of M collisions, independent of whether a collision causes a reaction or not. Using M collisions (a generation) as a unit of time is appropriate because otherwise an increase of the reactor size M would result in a slow down of development speed. If, for example, in a reactor with a population size of 10^5, 10^{10} collisions were computed, this would mean that, statistically, every molecule of the population did participate twice in a reaction. If the same number of collisions were computed in a reactor of size 10^{20}, only half the molecules would have taken part in a reaction. This would result in a fourfold slower development, if time would not be scaled by reactor size.

In continuously modeled ACs the progress of time depends on the integration step size of the numerical ODE solver. For a standard solver, a fixed step length of $h = 10^{-2}$ is a common setting. Thus, 100 evaluations of the ODE are needed to get a unit length time progress.

2.3.8 Pattern Matching

Pattern matching is a method widely used in artificial chemistries and other artificial life systems. A pattern can be regarded as information about (or as a means to identify) the semantics of a subcomponent or location. It allows to refer to parts of a system in an associative way independent from the absolute position of these parts. Koza for example used the shape of trees to address subtrees and to select reaction partners [77]. In the field of DNA computing, pattern matching is a central mechanism in the alignment of two DNA strands. There, a pattern is a sequence of nucleotides such as $CGATTGAGGGA\ldots$ In TIERRA, a pattern is given by a sequence of NOP0 and NOP1 operations in the genome of an individual and is used to direct jumps [107] of the program counter.

The accuracy of a match can be used to calculate the probability of a reaction or rate constants, as suggested in, e.g., [9]. In order to compute the accuracy, a distance measure between patterns is needed, that computes the "similarity" of two molecules by assigning a numerical value. A convenient distance measure for variable length sequences is the *string edit distance*. The distance between strings of the same length is usually computed by the *Hamming-distance*.

[7] This can be circumvented by using a deterministic pseudo random number generator trading space requirements with time requirements.

2.3.9 Spatial Topology

A spatial structure of the reactor is a parameter of the algorithm A of the AC $\{S, R, A\}$ and has no further influence on S and R except for some approaches which allow molecules to grow and occupy more than one point in the reactor space (see section 3.7 for a description of such ACs). In most cases, the reactor is modeled as a well-stirred tank reactor, often with a continuous inflow and outflow of substances. In a well-stirred tank reactor the probability for a molecule s_i to participate in a reaction r is independent of its position in the reactor. In a reactor with topology, the probability depends on the *neighborhood* of s_i. This neighborhood may be determined by the vicinity of s_i in an Euclidian space. This space may be two- or more-dimensional. Also, the neighborhood may be defined as the neighborhood in a cellular automaton (e.g., [87, 131]) or as a self-organizing associative space ([41]).

All these different spatial structures have one thing in common: they all have a more or less important influence on the selection of reactands, performed by the algorithm A. Adopting an additional spatial structure, the analysis of ACs becomes a more complicated task, but on the other hand, some of the observed properties might not occur without.

Ab hier wieder alles wie vorher.

3 Approaches

In this section methods for building artificial chemistries are reviewed in some detail. They are organized according to their underlying interaction mechanism and the structure of their molecules.

3.1 Rewriting or Production Systems

A *rewriting system* consists of certain entities or symbols and a set of syntactic rules for performing replacements. A rule defines whether a pattern of symbols can be replaced by another pattern. There are two variants for utilizing rewriting systems for defining the reactions. On the one hand the application of a rule corresponds to a reaction (see Secs. 3.1.1-3.1.3), on the other hand one single reaction consists of multiple applications of many rules (see Sec. 3.1.4 and [117]). Rewriting systems are also used to model chemical computation [30, 100].

3.1.1 The Chemical Abstract Machine (CHAM)

Berry and Boudol have developed an abstract machine, called *chemical abstract machine* (CHAM) which is based on the chemical information processing metaphor to model concurrent computation [20]. The machine is an extension of the Γ language introduced by Benâtre and Le Métayer [68].

Molecules: The molecules $s_1, s_2, \ldots$ are terms of an algebra. The **reactor** (population, solution) is a finite multiset which may contain molecules as well as subpopulations called subsolutions. Subpopulations are treated like molecules. Reactions may take place in subpopulations independently and in parallel. A membrane concept enables molecules to leave subpopulations.

Reactions: A special CHAM is defined by a set of **transformation laws**. There are four *general laws* that are valid for every CHAM and are the only rules that contain premises[8]. The *specific rules* define a special CHAM and are simple rewriting rules of the form

$$s_1, s_2, \ldots, s_k \longrightarrow s'_1, s'_2, \ldots s'_l \tag{6}$$

The general laws define how these rewriting rules can be applied to a multiset. Roughly speaking they say that molecules inside a (sub-)population that matches the left side may be replaced by the molecules on the right side (reaction and chemical law), that a subpopulation may develop independently (membrane law), and that molecules may enter or leave a subpopulation (airlock law).

To define a special CHAM an arbitrary set of special rules of the form (6) can be added to the general laws. To ease handling of the special rules Berry and Boudol distinguish three classes of special rules, namely, heating rules, cooling rules, and reaction rules. A *heating rule* decomposes a molecule into its components. A *cooling rule* is the corresponding counterpart. Heating and cooling is considered to be reversible. *Reaction rules* are

[8] These transformation rules are rules for a formal production system and should not be confused with reaction rules of artificial chemistries.

irreversible and are usually applied to heated molecules. They change the information content of a population [20].

Dynamics: At each time a CHAM may perform an arbitrary number of transformations in parallel provided that no molecule is used more than once to match the left side of a reaction law. A CHAM is non-deterministic if more than one transformation rule may be applied to the population at a time. In this case the CHAM selects a transformation rule randomly. (This selection process is guided by an objective function in the chemical casting model by Kanada et al. Sec. 3.1.3)

3.1.2 The Chemical Rewriting System on Multisets (ARMS)

The chemical abstract machine has been developed to model concurrent computing. However Suzuki and Tanaka demonstrate that it can be utilized to model chemical systems [120, 121]. They defined an ordered abstract rewriting system on multisets called *chemical ARMS*.

Molecules: Similar to a CHAM, the molecules are abstract symbols.

Reactions: The reaction rules are rewriting rules.

Dynamics: The reactor is represented by a multiset of symbols. In addition there is a set of input strings and an optional rule order. The *rule order* specifies in which order the rules are processed. It is used to analyze the influence of the order of rule application on the dynamics. Different rate constants are modeled by different frequencies of rule application.

It is shown that ARMS is able to model oscillating chemical systems, such as the Brusselator. Suzuki and Tanaka investigated the qualitative dynamics of ARMS by generating rewriting rules randomly and derived criteria for the emergence of cycles [120]. They also defined an order parameter λ_e which is roughly the relation of the number of heating rules to the number of cooling rules [121]. They found that for small and for large λ_e values the dynamics is simple, i.e. the rewriting system terminates and no cycles appear. For intermediate λ_e values cycles emerge. Recently ARMS has been applied to model ecological systems (see Sec. 4.1.3).

3.1.3 The Chemical Casting Model (CCM)

The chemical casting model (CCM) introduced by Kanada is also a production system inspired by the chemical metaphor. The motivation was to develop a new method for optimization of constraint satisfaction problems. In addition the CCM can be regarded as a model for emergent computation.

Molecules: Molecules (composed of atoms) and reaction rules depend on the type of the search problem. In general, atoms are components of a potential solution of the problem. In the examples shown in [71, 70] the whole population (working memory) represents one point in search space. Each *atom* has a type and a state. Atoms can be connected by *links*. Links may be used to specify a concrete problem like in the graph coloring example. In this case links are fixed during a simulation. For a traveling salesman problem (TSP) links can represent a solution to the problem and thus have to change during the search process.

Reactions: The reaction rules have in general the form

$$LHS \longrightarrow RHS \tag{7}$$

where the left hand side (LHS) is a sequence of patterns. If a subset of atoms matches the LHS it may be replaced by atoms which match the right hand side (RHS).

Dynamics: The quality of a solution is measured by a local evaluation function called *local order degree* (LOD). The LOD is used to decide whether a rule can be applied or not. There are two types of LODs: The first one maps one atom to a number and the second one maps two atoms to a number. A rule may be applied if the sum of the LODs of the participating atoms on the left hand side is smaller than the sum of the LODs on right hand side. The quality of the solution is only measured locally by this method.

In [70] Kanada has also introduced an annealing method by assigning an additional energy state variable, called *frustration* to each atom. The frustration is similar to a kinetic energy. The frustration is added to the LOD so that, if the frustration of atoms is high they may react even if the previous LOD criteria is not fulfilled. In case of an elastic collision the frustration of the colliding atoms is increased by a factor c. In case of a reaction, the frustration is reset to an initial value.

Kanada and Hirokawa have demonstrated successfully the application of the CCM to different constraint satisfaction problems, namely, graph coloring, N-queens problem, and TSP. The simulations show that even if during the simulation the current solution is evaluated only locally, the system can converge to the global optimum. They have also shown how the locality of the reaction rules can be tuned by adding a catalyst. Adding a catalyst reduces the locality of a rule which roughly leads to faster convergence speed but increases the probability that the system get stuck in a local optimum. The advantage of this approach is that many processors may operate on one instance of a problem solution in parallel with minimal communication because every evaluation of the quality and every variation is performed locally. In some cases (like graph coloring) even synchronization and access conflict to a shared memory can be ignored, because they only generate a harmless noise of low intensity. A problem is the termination criterion. In general it is a global operation to determine whether the optimum is found (if possible). A local termination criterion would have to estimate the global quality which results in a longer computation time.

3.1.4 Lambda-Calculus (AlChemy)

The λ-calculus provides a way to define an implicit structure-function mapping. It has been used by Fontana to define a constructive artificial chemistry [49].

Molecules: The set of molecules called *model universe* ([50], p. 10) are normalized λ-expressions. A λ-expression is a word over an alphabet $A = \{\lambda, ., (,)\} \cup V$ where $V = \{x_1, x_2, \dots\}$ is an infinite set of available variable names. The set of λ expressions S is defined for $x \in V, s_1 \in S, s_2 \in S$ by

$$\begin{aligned} x &\in S && \text{variable name} \\ \lambda x.s_2 &\in S && \text{abstraction} \\ (s_2)s_1 &\in S && \text{application} \end{aligned}$$

An abstraction $\lambda x.s_2$ can be interpreted as a function definition, where x is the parameter in the "body" s_2. The expression $(s_2)s_1$ can be interpreted as the application of s_2 on s_1. This can be formalized by the β-rule:

$$(\lambda x.s_2)s_1 = s_2[x \longleftarrow s_1] \tag{8}$$

where $s_2[x \longleftarrow s_1]$ denotes the term which is generated by replacing every unbounded occurrence of x in s_2 by s_1. A variable x is bounded if it appears in in a form like $\ldots(\lambda x.\ldots x\ldots)\ldots$. It is also not allowed to apply the β-rule if a variable becomes bounded. Example: Let $s_1 = \lambda x_1.(x_1)\lambda x_2.x_2$ and $s_2 = \lambda x_3.x_3$ then we can derive:

$$(s_1)s_2 \Longrightarrow (\lambda x_1.(x_1)\lambda x_2.x_2)\lambda x_3.x_3 \Longrightarrow (\lambda x_3.x_3)\lambda x_2.x_2 \Longrightarrow \lambda x_2.x_2 \tag{9}$$

Reactions: The simplest way to define a reaction scheme for two colliding molecules s_1 and s_2 is to apply s_1 to s_2:

$$s_1 + s_2 \longrightarrow s_1 + s_2 + normalForm((s_1)s_2) \tag{10}$$

The procedure $normalForm$ reduces its argument term to normal form[9]. The λ-calculus allows a generalization of the collision rule by defining it by λ-expression $\Phi \in S$:

$$s_1 + s_2 \longrightarrow s_1 + s_2 + normalForm(((\Phi)s_1)s_2) \tag{11}$$

Dynamics: Fontana and Buss have performed a large number of experiments based on a λ-chemistry. In their experiments typically a simulation was used with explicit molecules and explicit single collisions according to the algorithm in Sec. 2.2.1, and a constant reactor size ($M = 1000-3000$). The reactor is initialized with random molecules.

With the simple reaction scheme and algorithm of the first example (Sec. 2) Fontana observed that the diversityof the population reduced quickly, often leading to only one surviving self-replicating species. The organization structure of the surviving ensembles is called *level-0-organization*. In general, a level-0 organization consists of a few closely coupled replicators.

In order to arrive at more complex reaction networks Fontana introduced additional filter conditions. For instance, a collision was considered to be elastic if the outcome of the reaction function was equal to one of the reactants. In such a case an exact replication was not allowed. The result of these additional filter conditions were reaction networks called *level-1 organizations* composed of a huge, even infinite, number of molecules. These organizations were very stable. Because the reactor size was small compared to the size of a level-1 organization, a level-1 organization was only represented by some if its species in the reactor and constantly produced new species (with respect to the species in the reactor). A randomly generated species inserted into a reactor containing a level-1 organization would be incorporated into the organization only rarely. If it was incorporated the new organization was typically extended in layers.

[9]The reduction process is bounded by a maximum of available time and memory. If these resources are exceeded before termination, the term is considered to be unstable and no reaction product results.

Fontana further defined *level-2 organizations* which consist of two (or more) co-existing level-1 organization. The stable co-existence is characterized by the following two conditions: 1.) The organization interact. 2.) The interaction is moderated by intermediate species called *glue*. A spontaneous emergence of a level-2 organization from a randomly initialized population is extremely seldom. However, it can be synthetically generated by merging two indepently evolved level-1 organization.

Fontana discovered that the species dominating the reactor after the transient phase have a similar syntactical structure [49]. This phenomenon is typical for constructive artificial chemistry and appears even in systems with totally different representation and reaction mechanisms. It is discussed in detail in Sec. 5.

3.2 Arithmetic Operations

3.2.1 Simple Arithmetic Operators

As we have already shown in the example of the number-division chemistry representations and operators can be borrowed from mathematics to construct artificial chemistries with interesting behavior. To demonstrate that even the simplest reaction rules may generate a (presumably) chaotic behavior, one can construct the following chemistry. **Molecules:** The molecules are natural numbers.

Reactions: Fig. 3 shows a typical simulation of a number-addition chemistry where the reaction mechanism is simply the integer addition operation:

$$s_1 + s_2 \longrightarrow s_1 + s_2 + reaction(s_1, s_2) \tag{12}$$

The function *reaction* is defined as $reaction(s_1, s_2) = s_1 + s_2 \mod 2^n$ with n the size of the integer representation, here $n = 32$.

Dynamics: Initializing the well stirred tank reactor with only 10 different numbers, the reactor behaves apparently totally chaotic after a very short transient phase. That is to say the content of the reactor looks as if it had been generated randomly.

3.2.2 Matrix-Multiplication Chemistry

One of us has introduced a binary string chemistry based on matrix multiplication [13, 15, 16]. Here, the central operation is the folding of a binary string into a matrix which then operates on another string by multiplication.

Molecules: The molecules are binary strings. Simulations have been performed with fixed length (4-bit, 9-bit, and 16-bit) and variable length systems.

Reactions: Assume a reaction $s_1 + s_2 \Longrightarrow s_3$. The general approach is:

1. **Fold s_1 a to matrix M**
   Example: $s_1 = (s_1^1, s_1^2, s_1^3, s_1^4)$

$$M = \begin{pmatrix} s_1^1 & s_1^2 \\ s_1^3 & s_1^4 \end{pmatrix} \tag{13}$$

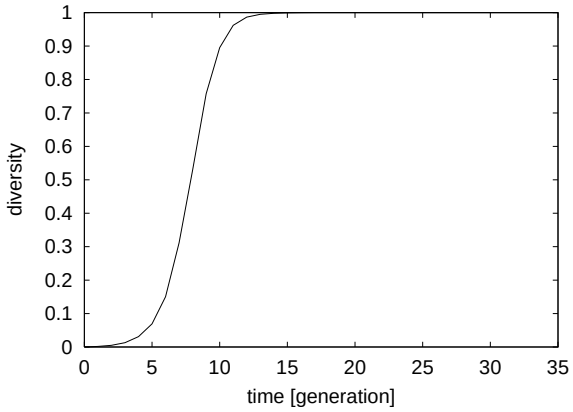


Figure 3: *A simulation of the number-addition chemistry. The population is initialized with one randomly generated molecular type. The figure shows the evolution of the relative number of different string types in the population. Population size* $M = 10000$*. Reaction* $s_3 = s_2 + s_1 \mod 2^{32}$*.*

2. **Multiply M with subsequences of s_2**
   Example: Let $s_2 = (s_2^1, s_2^2, s_2^3, s_2^4)$ be divided into two subsequences $s_2^{12} = (s_2^1, s_2^2)$ and $s_2^{34} = (s_2^3, s_2^4)$. Then we can multiply M with the subsequences:

$$s_3^{12} = M \odot s_2^{12}, \quad s_3^{34} = M \odot s_2^{34} \tag{14}$$

3. **Compose s_3 by concatenating the products.**
   Example: $s_3 = s_3^{12} \oplus s_3^{34}$

There are various ways of defining the vector matrix product $\odot$. It was mainly used with the following *threshold multiplication*. Given a bit vector $x = (x_1, \ldots, x_n)$ and a bit matrix $M = (M_{ij})$ then the term $y = M \odot x$ is defined by:

$$y_j = \begin{cases} 0 & \text{if } \sum_{i=1}^{n} x_i M_{i,j} \leq \Phi \ , \\ 1 & \text{otherwise.} \end{cases} \tag{15}$$

The threshold multiplication is similar to the common matrix-vector product, except that the result vector is mapped to a binary vector by using the threshold Φ. For the 4-bit example the threshold has been usually set to $\Phi = 0$.

Dynamics: Simulations have been performed with implicit as well as with explicit molecules and collisions.

The general result was that such a system would quickly develop into a steady state where some string species support each other in production and thus become a stable autocatalytic cycle, whereas most others would disappear due to the competition in the reactor. Small systems could also be modelled by differential rate equations and were in good agreement with the simulation of explicit and implicit molecules. The complexity of interactions even of small systems, say of bit string length 4, was surprising [14]. Figure 4 shows a metabolism developing in one of the smallest non-trivial systems of this kind.

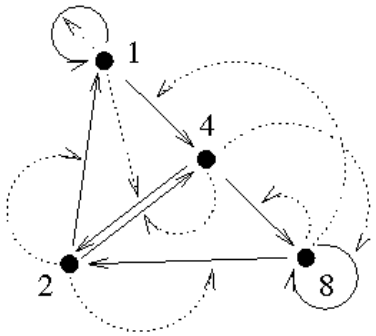


Figure 4: *Reaction graph of the metabolism of $N = 4$. Taken from [14].*

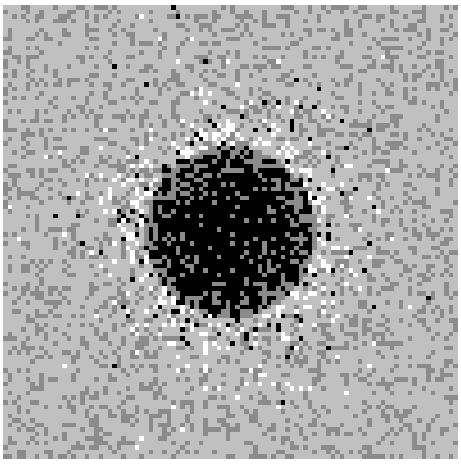


Figure 5: *Stable membrane-like structure of $N = 4$ matrix multiplication AC with a topology. Taken from [17].*

In later work, the matrix-multiplication system was equipped with a topology. This allowed an even larger set of interactions. Notably, string species that would die out in an unbounded interaction space could survive in the neighborhood of other string types. As a result, membrane-like structures formed in a $N = 4$ matrix-multiplication chemistry [17]. Figure 5 shows one of these structures from a simulation.

3.3 Autocatalytic Polymer Chemistries

Bagley, Farmer, Fontana, and Kauffman et al. [9, 10, 48, 73, 74, 86] have studied the emergence and evolution of autocatalytic metabolic networks by using a constructive artificial chemistry, where the molecules are character sequences and the reactions are concatenation and cleavage. They have shown how small, spontaneous fluctuations can be amplified by an autocatalytic network, possibly leading to a modification of the entire network. A sequence of such modifications has been identified as an evolutionary process. The spontaneous fluctuations are explicitly induced by the meta-dynamical algorithm which can be interpreted as explicit (passive) mutation. The modifications of the reaction network are achieved by means of randomly modifying reaction sets [9, 10, 74] or by

evolving reaction networks with an evolutionary algorithm [86]. Bagley, Farmer et. al mentioned experiments in which they modified the reaction network according to the present polymers and their structure, but their main interest was on random catalytic networks.

Artificial polymer chemistries are intentionally motivated by the investigation of the emergence of autocatalytic sets of proteins [73], which, together with the theory of hypercycles [47] and prebiotic molecular evolution [75, 101] are considered to be essential for the origin of protein based organic life. Random reaction networks, analyzed by [72, 76] and others were additionally helpful.

Molecules: All polymer chemistry approaches make use of molecules which are character sequences over a finite set $S = \{a, b, \dots\}$ of so called *monomeres*.

Reactions: In most of the cases, the reactions are of the type

$$s_1 + s_2 + s_3 \rightleftharpoons s_1 s_2 + s_3 \tag{16}$$

which are catalyzed concatenation and cleavage reactions.

Dynamics: The experiments use a well stirred tank reactor without any topological structure which is simulated by a meta-dynamical ODE frame work.

An interesting aspect is that the catalytic or autocatalytic polymer sets (or, reaction networks) evolve without having a genome. The inheritance of information in these sets and the question when and how the transition from genome-less evolution and inheritance of information to genome-based replication took place is discussed by Kauffman in [73, 74].

Bagley, Farmer and Fontana have also studied the emergence and evolution of autocatalytic metabolic networks by using a constructive artificial chemistry, where the molecules are character sequences and the reactions are concatenation and cleavage [10]. They have shown how small, spontaneous fluctuations can be amplified by an autocatalytic network, possibly leading to a modification of the entire network. A sequence of such modifications has been identified as an evolutionary process. The spontaneous fluctuations are explicitly induced by the meta-dynamical algorithm which can be interpreted as explicit (passive) mutation.

3.4 Abstract Automata

The concept of an abstract automaton or machine commonly used in computer science provides a variety of approaches to define structure-function relationships. Interesting candidates are finite state machines [40] or Turing machines [66, 129] which are operating on binary data. So, it is quite natural to represent the molecules as collections of bits organized as binary strings. The state transition function of the machines can be represented as a lookup table or by a program (a sequence of commands). A molecule appears in two forms: 1.) As passive data (binary string) and 2.) as an active machine. There is a mapping from a binary string into its machine form called *folding*. The folding may be indeterministic and may depend on other molecules (e.g., [66]).

3.5 Artificial Molecular Machines

Already in the early 70s Laing argued for abstract, non-analogous models in order to develop a general theory for living systems [78]. For developing such a general theory so called *artificial organisms* would be required. Laing suggested a series of artificial organisms [78, 79, 80, 81, 82] that should allow to study general properties of life and thus would allow to derive a theory which is not restricted to the instance of life we observe on earth. The artificial organisms consist of different compartments, e.g., a "brain" plus "body" parts [79]. These compartments contain molecules from an artificial chemistry.

Molecules: The molecules are strings of symbols, usually binary strings. Molecules can appear in two forms. They can be binary data strings or machines. In machine form the molecules are represented as a sequence of instructions where the sequence possesses a three-dimensional shape. So, the machine form is a molecule which contains loops. The motivation design is to avoid jumps which are biologically implausible according to Laing [80].

Reactions: In order to perform a reaction two molecules are attached to each other such that they touch at one position (atom). One of the molecules is considered as the active machine which is able to manipulate the passive data molecule. For running the machine the instruction at the binding position is executed. Depending on the content of data and machine molecules at the binding position, the binding position is moved to the left or right on the data molecule. On the machine molecule the binding position moves also which is equivalent to pointing to the next instruction.

Dynamics: Laing has not considered an explicit dynamics, probably because a dynamical simulation of his artificial chemistries has been beyond the computational means available back then. However, he proved that his artificial organisms are able to perform universal computation [79] and he also demonstrated different forms of self-reproduction, self-description and self-inspection [80, 81, 82].

3.5.1 Polymers as Turing Machines

McCascill investigated the self-organization of the genetic apparatus by a model reactive polymer chemistry implemented in silico. Starting from a implementation as software [91], he realized the AC in a specially designed parallel computer [28, 46, 127]. The early work by McCascill should be described in more detail:

Molecules: Linear macromolecular heteropolymers provide a simple class of molecules with combinatorial complexity. Polymers consisting of finite sequences of bases from a finite set of monomers (restricted to two) with a bounded length l act as the set S. In earlier experiments, l was typically 20, in later experiments with special parallel hardware, the length was not limited any more. Each molecular string can be processed as a linear molecule and also codes for an automaton. This is a well known property of universal Turing machines which also divide the tape into program and data.

Reactions: Binding dynamics is integrated efficiently using a pattern matching collision table. This results in a molecular pattern-based interaction where the relationship between function (here: replication) and sequence is externally specified. Only bimolec-

ular or unimolecular reactions occur, with all unimolecular rates set to zero or to one[10]. To encode the relationship between sequences, the two-body-interaction scheme uses patterns $\{0, 1, \#\}^*$ (# is a "don't care" symbol). The number of "#" 's in the condition (pattern) determines the specificity of the recognition. Strings of length l encode $2^{m+1} - 1$ different patterns of length up to $m = 1, 2, \ldots, l$. Only rules for the derivation of kinetic parameters from macromolecular sequences are given: the bimolecular rates thus depend on concentration and specificity. The general reaction scheme can be written as:

$$s_1 + s_2 \longrightarrow s_1 + s_3, \text{ if } p_1 \subseteq s_1 \text{ matches } p_2 \subseteq s_2. \tag{17}$$

In this reaction, p_1 and p_2 are substrings (patterns) of the polymers, p_1 is the processor and p_2 acts as the data.

Dynamics: The reactor realizes a two-dimensional domain (in the early experiments a linear domain) in which the polymers are set to random locations. In the latter case, the neighborhood is defined by a relationship in "pattern-space", in the former case the neighborhood is defined by the location in the Euclidian space of the reactor.

In later work [28, 127] ACs were implemented and simulated with reconfigurable hardware based on FPGAs. The algorithm realizes a "pattern processing chemistry" and this processing is potentially universal: it is a probabilistic and constructive variant of a Turing machine operating on a polymer tape. A special error rate parameter controls the precision of elementary steps of the Turing machine. Without this parameter, the dynamics would reduce to a Markovian dynamics: the population of sequences at time t would determine the future dynamics. This error rate can be regarded as "mutation". The real encoding of processors from crossed molecular strings is a modification of a Turing machine: it has two tapes and no finite state machine. The point of intersection moves, symbols under the point of intersection being directly modified, and the direction of future motion is determined by the two symbols. The tapes of the Turing machine are built with two randomly chosen polymers, with the first string acting as processor and the second string acting as data. The resulting two strings are returned to the population.

It is observed that during the development of the system strings with the ability to self-replicate appear. In coevolution with parasites an evolutionary arms race started among these species and the self-replicating string diversified to an extent that the parasites could not coadapt and went extinct. In the lattice environment, sets of cooperating polymers evolved, interacting in a *hypercyclic* fashion. In the latest experiments, a *Chemoton*-like cooperative behavior appeared, with spatially isolated, membrane-bounded evolutionary stable molecular organizations. These organizations were able of self-assembling their surrounding membrane and, as a whole, "diffused" in space. Spatial isolation allows the diversification into separated competing species (each representing open reaction networks). Cooperative organizations show an evolutionary advantage over more "egoistic" organizations.

3.5.2 Machine-Tape Interaction

Ikegami and Hashimoto developed an abstract artificial chemistry based on binary strings where the interaction is defined by a Turing machine with finite circular tape.

[10] A reaction rate of zero indicates an elastic reaction.

Molecules: There are two molecular types: tapes and machines. Tapes are circular binary strings (in [66] 7 bits long). Machines are binary string composed of a head (4 bits) a tail (4 bits) and a state transition lookup table (8 bits). Tapes and machines form two separate populations in the reactor.

Reactions: Reactions take place between a tape and a machine according to the following reaction scheme:

$$s_M + s_T \longrightarrow s_M + s_T + s'_M + s'_T \tag{18}$$

A machine s_M is allowed to react with a tape if its head matches a substring of the tape s_T and its tail matches a different substring of the tape s_T. The machine operates only between these two substrings (called reading frame) which results in a tape s'_T. The tape s'_T is folded (translated [66]) into a machine s'_M by reading 16 bits in circular way starting at the head binding position. Thus, a tape s'_T can be translated into different machines if there are machines that bind to different positions at tape s_T.

Dynamics: The molecules are represented implicitly by an integer frequency vector. During the simulation the population sizes of machines and tapes are kept constant, like in [52, 40]. An update is performed by replacing a fraction of c (in [66] $c = 60\%$) molecules of the reactor by reaction products. The composition of the reaction products is calculated using rate equations with real valued variables and parameters. For this, the integer frequency vector is converted into a real valued concentration vector and the result of the rate equations is rounded to obtain again integer values.

Ikagami and Hashimoto showed that under the influence of low noise simple autocatalytic loops are formed. When the noise level is increased, the reaction network is destabilized by parasites, but after a relative long transient phase (about 1000 generations) a very stable, dense reaction network, called *core network*, [66] is formed. A core network maintains its relative high diversity even if the noise is deactivated. The active mutation rate is high. When the noise level is very high, only small, degenerated core networks emerge with a low diversity and very low (even no) active mutation.

At a low noise level reaction cycles are formed which are similar to hypercyclic organizations proposed by Eigen and Schuster [47]. Ikegami and Hashimoto called them Eigen/Schuster type networks which have the following form:

$$\begin{aligned} s_M + s_T &\longrightarrow s_M + s_T + s'_M + s_T \\ s'_M + s'_T &\longrightarrow s'_M + s'_T + s_M + s'_T \end{aligned} \tag{19}$$

A reaction network like Eq. (19) does not perform any active mutation.

At higher noise level different type of reaction cycle emerged, called *double autocatalytic network* [66]. These cycles maintain a high active mutation and do not perform in their pure form exact replication. The following network is an example for a double loop type network:

$$\begin{aligned} s_M + s_T &\longrightarrow s_M + s_T + s'_M + s'_T \\ s'_M + s'_T &\longrightarrow s'_M + s'_T + s_M + s_T \end{aligned} \tag{20}$$

The core networks which emerged under the influence of external noise are very stable so that there is no further development after they have appeared. To promote evolution

reaction systems can be encapsulated into cells (compartments) where substances are allowed to diffuse. Ikegami and Hashimoto showed that this can result in complex, co-evolutive behavior [67].

3.5.3 Automata Reaction

The automata reaction, developed by us [40], is based on a formal finite state automaton which is a mixture of a Turing-machine and a command driven register machine. It has been inspired by Typogenetics [62]. The automata reaction has been designed in consideration of "respecting the medium" [107], here a von Neumann computer, so that it is fast and efficient which allows large population sizes (e.g., 10^6) and long simulations (e.g., 10.000 generations).

Molecules: The molecules are fixed length binary strings $S = \{0,1\}^{32}$. This allows easy and efficient explicit simulations.

Reactions: An interaction of the form

$$s_1 + s_2 \longrightarrow s_1 + s_2 + s_3 \tag{21}$$

among two strings $s_1, s_2 \in S$ is performed in two steps: (1) s_1 is mapped to a finite state automaton A_{s_1} by interpreting s_1 as 4-bit machine code. (2) The automaton A_{s_1} is applied to s_2 to generate the output s_3. Note that after the reaction the automaton is discarded and s_1 is not changed. Each sequence may act as an automaton and/or be processed as an automaton with equal probability. There is no population of automata like in the system by Ikegami and Hashimoto.

Figure 6 shows the structure of the automaton. It contains two 32-bit registers, the *IO register* and the *operator register*. At the beginning operator string s_1 is written into the operator register and operand s_2 into the IO register. The program is generated from s_1 by simply mapping successive 4-bit segments into instructions. The resulting program is executed sequentially, starting with the first instruction. The automaton halts after the last instruction has been executed. During program execution the automaton can only modify bits in the IO register, but read bits from the operator register. There are logic and arithmetic operations which take as input two bits, one from the IO and one from the operator register, respectively. The result is stored in the IO register. The position of processed bits is under the control of the program. In addition in some experiments elastic collisions are introduced by not allowing exact replications [49].

Dynamics: With the automata reaction several experiments have been performed using explicit single molecule collisions and explicit populations with and without a topological structure. Typical phenomena which have been observed are: dominance of self-replicators, the formation of catalytic networks, complexity increase when exact replication is disallowed and syntactic similarity of molecules forming the emergent organizations. These observations support the findings of Fontana et al., especially the phenomenon of syntactic and semantic closure. In large systems of this AC, evolutionary phenomena have been observed. This is notable because no explicit variation (passive mutation) and no explicit fitness function is present in the system. Variation and "natural" selection is driven by the molecules themselves. For this reason the phenomenon has been termed

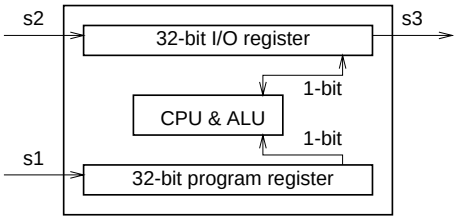


Figure 6: *A schematic diagram of the automata reaction. The CPU executes the program specified by s_1. Roughly, 4-bits are interpreted as one command. The CPU may read the program and I/O register but can only write the I/O register. With an appropriate program the contents of the program register can be copied to the I/O register.*

self-evolution. Further phenomena observed are spontaneous emergence of recombination and spontaneous transition from lazy to eager replicators [11]

3.6 Assembler Automata

An *Assembler automaton*[12] is a parallel computational machine which consists of a core memory and parallel processing units. It can be regarded as a parallel von Neumann machine. Assembler automata have been used to create certain artificial life systems. In this section we will describe the basic functionality of assembler automata and discuss their relation to artificial chemistries.

The first implemented assembler automaton of competing computer programs is Core War. Core War has been design by Dewdney as a game where handwritten computer programs struggle for computer resources and may fight each other [39]. The system is in fact a multi-processor, shared-memory system. All processors operate on the same linear and cyclic memory called *core*. The machine code (called redcode) consists of only 10 instructions with one or two arguments. Memory cells can be addressed only direct-relative or indirect relative. Thus, the execution of a program is independent from its absolute position in the memory.

For a typical match two programs which should play against each other are stored at random locations in the core memory. Then they are started in parallel by assigning each program a CPU with its own program counter (PC) and registers. The following example illustrates the redcode language. It is a program writing a zero at every 5th memory location which should overwrite and destroy code of its opponent.

cell no.	instruction	arg1	arg2	comment
0	DAT		-1	memory cell used as a variable which is initialized by -1
1	ADD	#5	-1	adds 5 to the contents of the previous memory cell
2	MOV	#0	-2	writes 0 to the memory cell addressed indirectly by the memory cell 0
3	JMP	-2		jump two cells backwards

[11] A *lazy replicator* s_1 copies every molecule it collides with and can mathematically viewed as an identity function: $s_1 + s_2 \Longrightarrow s_2$. An *eager replicator* s_1 creates a copy of itself: $s_1 + s_2 \Longrightarrow s_1$.

[12] The term *assembler automata* has been coined by Rasmussen.

The simplest self-replicating program consists of only one instruction: MOV 0 1. This instruction copies the contents of the relative address 0 (the MOV 0 1 instruction) to the next memory cell (relative address 1). Thus, the program scrolls through the memory leaving a track of MOV 1 0 instructions behind. There is also an instruction to split execution which creates a new process.

3.6.1 Coreworld

Rasmussen et al. have used Core Wars to build an artificial chemistry called *Coreworld* [102, 103]. They introduced *random fluctuations* in two different ways: 1.) New processes are started at random with randomly initialized PCs. 2.) When a MOV instruction is executed the copied data may be mutated. In addition *computational resources* are introduced explicitly. Each cell holds an amount of computational resources (a number) which is used up by program execution and which is refilled by a constant influx. There is also a maximum *operation radius* which restricts relative addressing and promotes locality.

Rasmussen et al. observed through various experiments that the qualitative dynamic behavior depends crucially on the parameter setting. For low resource influx (called *desert condition* only simple cyclic structures dominated the converged population. For high resource influx (called *jungle condition*) diversity becomes much richer and more complex even cooperating structures emerge. **Molecules:** A problem in Coreworld is to identify a molecule or organism which might be defined as a collection of cells or instructions. The smaller an individual is the harder can one distinguish it from the environment. In addition situations are difficult where instruction pointers are intermingled. Ray has overcome this problem by introducing a memory management in Tierra.

Reactions: The Coreworld chemistry is different from other ACs because there are no explicitely defined reactions between molecules. Due to the fact that it is difficul to identify molecules, interactions between molecules are even more difficult to determine. Implicitely, a reaction might be seen as the execution of code belonging to another molecule.

Dynamics: The dynamics of the system is achieved by many CPUs working in parallel on different pieces of code. The code moves within the memory and changes due to explicit mutations or due to changes caused by other programms.

3.6.2 Tierra

Ray has designed *Tierra* to model the origin of diversity of life (e.g., the Cambrian explosion) not its origin [107]. Tierra is similar to Core War with the following important modifications and extensions:

Small instruction set without numeric operands. The instruction set of Tierra consists of 32 instructions. Thus, only 5 bits are needed per cell. Therefore memory cells in Tierra are much smaller than in Core War where in addition to the opcode of the instruction the arguments have to be stored.

Addressing by templates. Memory locations are addressed by templates which consists of NOP0 and NOP1 instructions. Consider for example the following program sequence consisting of four instructions: JMP NOP0 NOP1 NOP0. The target for the

jump instruction is the closest of the inverse patterns formed by the three instructions NOP1 NOP0 NOP1.

Memory allocation and protection. Programs can allocate memory which is protected by a global memory manager. Other programs are not allowed to write protected memory but may read or execute it. By this method is much easier to specify and intuitive clear which memory cells belong to a certain organism.

Artificial fitness pressure. Ray also introduced an artificial, explicit fitness pressure by penalizing "illegal" or unwanted operations; like a JMP instruction where no target pattern can be found or division by zero. Programs executing many unwanted operations are more likely to be terminated by a reaper process which is invoked when the allocated memory exceeds a predefined threshold. This mechanism allows also to create a fitness pressure towards more efficient organisms, easily.

Molecules, reactions, and dynamics are nearly the same as in Coreworld, exept for the "molecules" which are easier to determine because they possess a clearly separated and protected piece of memory. In typical experiments with Tierra the core memory is initialized with a handwritten program, the *ancestor*, which is able to self-replicate. Under the influence of mutations and random fluctuations the population diversifies. Ray has observed the evolution of parasites, protection against parasitism, hyper-parasites and cooperative organizations. It should be noted that the different species are similar to the ancestor and are often generated from it by a few point mutations, only.

It has been experimentally shown that in Tierra self-replicating entities, like the ancestor, do not appear spontaneously [105]. *Pargellis* has shown that this can be achived by simplifying the instruction set so that the probability to get a self-replicating program randomly is increased [98, 99] (s. also Primordial Soup by M. De Groot).

3.6.3 Avida

Ray's Tierra was further developed by Adami's **Avida** group. The major difference between Tierra and Avida is the restriction to only local interaction between individuals and the maximum population size given by the dimensions of the grid and not by the size of the total genome space of the population.

Molecules: A molecule is single assembler program. The assembler language is a reduced variant of Tierras instruction set. Initial molecules are usually hand-written self-replicating programs.

Reactions: In the fundamental Avida system only unimolecular first-order reactions occur which are of replicator type. The reaction scheme can be written as

$$s \xrightarrow{E} s' + s \tag{22}$$

and thus are more similar to chemical interactions than in other assembler automata. In Eq. 22, $s' = s$ if no mutation occurs. The program s' is created by execution of s and explicit mutation of its output. E can be seen as a certain energy that is consumed during the replication and is modeled as CPU-time.

Dynamics: The population consists of a 2-D lattice where each lattice site holds one molecule (program). Programs are executed in parallel and independently. The system

is seeded with an ancestor and evolves by adapting the population via mutation to an external fitness-landscape that defines the amount of CPU time (i.e. the energy) given to an individual. Competition and selection occurs because programs replicate different speed depending on their code and energy consumption.

The *avida* group also investigated evolutionary learning, the emergence of gene expression, and similarities to population dynamics in *E. Coli* bacteria [4, 3].

3.7 Lattice Molecular Systems

In this section systems are discussed which consist of a regular lattice where each lattice site can hold a part (e.g., atom) of a molecule. Between parts bonds can be formed so that a molecule covers many lattice sites. This is different to systems where a lattice site holds a complete molecule, like in Avida or [17]. The important difference is that in lattice molecular systems the space of the molecular structure is identical with the space in which the molecules are floating around. In systems where a molecule covers just one lattice site the molecular structure is described in a different space independently from the space the molecule is located in.

3.7.1 Autopoietic System

Varela, Maturana, and Uribe introduced in [131] a lattice molecular system to illustrate their concept of autopoiesis which has also been investigated by Zeleny [138], McMullin and Varela [93], and others. The system consists of a 2-D squared lattice.

Molecules: Each lattice site can be occupied by one of the following **atoms**: Substrate S, katalyst K, and monomer L (also called link particle [93]). Atoms may form bonds and thus form molecular structures on the lattice.

Reactions: If molecules come close to each other they may react according to the following reaction rules:

$K + 2S \longrightarrow K + L$	(1) Composition: Formation of a monomer.
... -L-L + L $\longrightarrow$ ... -L-L-L	(2) Concatenation:
L + L $\longrightarrow$ L-L	Formation of a bond between a monomer and another monomer with no more than one bond. This reaction is inhibited by a double-bounded monomer.
$L \longrightarrow 2S$	(3) Disintegration: Decomposition of a monoer.

The inhibition of the reaction (2) by a double-bounded monomer has been reconsidered and identified as a crucial mechanism by McMullin and Varela [93].

Dynamics: The reaction rules are applied to lattice sites asynchronously similar to an asynchronous cellular automaton. Figure 7 illustrates an autopoietic entity which may arise in such an artificial chemistry.

This autopoietic entity has a cell-like structure. Its membrane is formed by a chain of L monomers and encloses one ore more catalysts K. Only substrate S may diffuse through the membrane. Substrate inside is catalyzed by K to form free monomers. If the membrane is damaged by disintegration of a monomer L it can be quickly repaired by a free monomer floating around inside the cell.

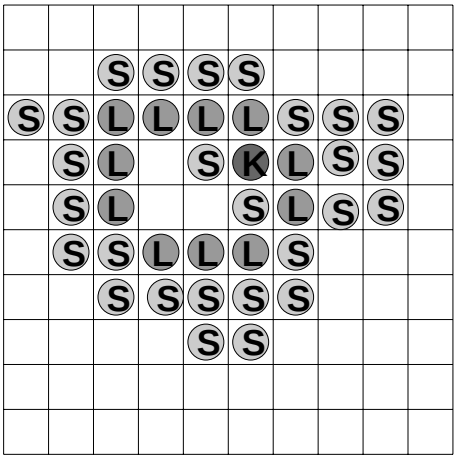


Figure 7: *Example of an autopoietic entity in a lattice molecular system by Varela, Maturana, and Uribe [131].*

A problem arises when monomers inside the cell form bonds among each other. In this case it is unlikely that the cell is able to self-repair [93]. Thus, a mechanism has to prevent the formation of bonds inside the cell. Here, this is achieved by the *chain based bond inhibition* which has already been used but not documented in the original work by Varela et al. [131] and reconsidered by McMullin and Varela [93].

They demonstrate by simulation that without chain based bond inhibition self-repair is unlikely, otherwise likely. But the simulation also showed that even with chain based bond inhibition the autopoietic entity is not asymptotically stable. In all reported 5 runs the autopoietic structure was able to sustain itself for about 200- 1500 updates cycles, only. There has been no systematically parameter studies performed, yet. So it can not be said yet whether the observed phenomena are typical and which necessary properties an artificial chemistry must have in order to allow persistent autopoietic structures.

3.7.2 Lattice Polymers

Lattice molecular system have been intensively used to model polymers [130], protein folding [116] and RNA structures. Beside approaches which intend to model real molecular dynamics as accurate as possible there are also approaches which try to build abstract models. These models should give insight into statistical properties of polymers, like their energy landscape and folding processes and how they are related to parameters such as primary structure and folding method. These abstract models may also be considered as artificial chemistries. For example there are also random chemistry approaches where interactions between monomers are chosen at random [109, 110].

We will not go into more details here, because the focus of these works is on the molecular dynamics (folding process) and statistical (thermodynamic) properties of polymer systems and not on fundamental questions concerning origin and evolution self-maintaining organizations or molecular information processing. For these questions a more complete system allowing diverse chemical (catalyzed) reactions and the simulation of larger spaces (reaction vessels) is needed which will be discussed in the following section.

3.7.3 Lattice Molecular Automaton (LMA)

Mayer, Köhler, and Rasmussen have developed the *lattice molecular automaton (LMA)* to enable simulation of large-scale molecular self-organization processes [89, 90, 94] The LMA is based on the lattice polymer automaton by Rasmussen and Smith [104], but is more realistic. The motivation of these automata is to give insight into the driving forces which are responsible for the origin and evolution of complex hierarchical structures and what is required for the emergence of these structures (minimal model).

The LMA is a synchronous, deterministic cellular automaton with discrete space, time, mass, and energy. Only integer arithmetic is used. The physical space is represented by a two-dimensional triangular lattice. Kinetics as well as potential energy is modeled but for the potential energy only intermolecular potentials are considered.

Molecules: Each lattice site may hold one atom or monomer (referred to as atom in the following) which is represented by a 7-tuple, called *data structure*. Three of these 7 components are used as temporary variables during one update cycle. An atom is described by its type, outgoing forces, local kinetic energy, and intramolecular bonds to other atoms in the neighborhood. Atoms can form polymers that can occupy several lattice sites. Atoms in a polymer are connected with a bond.

Reactions: Due to the fact that neither new bonds are created nor old bonds deleted, there are no reactions taking place. There still is interaction between molecules, on the level of forces, spatial movement and several kinds of energy

Dynamics: The synchronous and deterministic update rules of the cellular automaton cause the dynamics of the system. An interesting feature of the LMA is that forces are modeled by *force particles* which are propagated during one update cycle to the temporary variables of neighboring lattice sites. To illustrate the mechanism the update cycle will be briefly summarized. It consists of 18 sub-steps which are organized in the following 6 steps:

Step 1: Propagation of molecular types and kinetic energy: Thermal diffusion is performed by redistributing kinetic energies to neighbors.

Step 2: Construction of force fields: Outgoing force particles are copied into the corresponding temporary ingoing force particle variables. They may be modulated depending on the molecular type.

Step 3: Calculation of potential energies: The calculation of the potential energies is based on a Coulomb term, where ingoing force particles are multiplied with corresponding outgoing force particles. A rotation to the right or left takes place if the local potential energy is reduced by this rotation. This step also handles induced charge displacement (dipole - induced dipole interaction), polarization (induced dipole - induced dipole interaction), and cooperative increasing strength of bonds. After step 3 the sum of forces effecting a molecular object are calculated for all directions.

Step 4: Calculating of move direction: The move direction is calculated based on classical mechanics by taking the current kinetic energy ("speed") and the final potential energy of step 3 ("forces") into account. If the lattice site in this direction

is already occupied by a molecular object, the object remains at its current location. To avoid bond-breaking a movement is only performed, if the molecular object does not leave the neighborhood of any of its neighbors it is connected to. This will significantly reduce the movement.

Step 5: Update of bonds in polymers: Bonds inside polymers are moved according to the molecular movement in step 4. There are no new bonds created or removed.

Step 6: Movement of molecules: The non-temporary values of the molecular molecules are copied to neighboring cells according their previously calculated move directions.

In [90, 94] The LMA is demonstrated by simulating (a) cluster formation of polar molecules and (b) phase separation of a mixture of water molecules and hydrophobe monomers [94], (c) polymers in a simulating polymer dynamics in a polar environment (e.g., water) [90, 94], and (d) emergence of self-reproducing micell-like structures [89]. The authors were able to show, that the measured radial distribution function [13] (RDF) of water corresponds qualitatively with the RDF in the simulated system (a). The simulated phase separation is similar to the observed process real mixtures (b). The hydrophobic effect (which is responsible for assembly of membranes of cells and intracellular compartments [126]) appeared as an emergent phenomenon. Polymers form higher order structures in LMA simulations such as the micell-like patterns shown in [89].

The formation of these micells can be considered as an emergent process with at least two hierarchical levels. The first level is the formation of hydrophobic, hydrophilic, or amphiphilic polymers and the second step the formation of larger micell-like clusters which consist of amphiphilic polymers. It should also be noted that these micells reproduce. A hydrophobic surface of a micell is able to catalyze hydrophobic polymers to become amphiphilic which are afterwards incorporated into the micell. The micell grows until it becomes unstable and "divides" [89].

A problem is fast propagation of information which is required for realistic movement of large molecules. If the updates are strictly performed locally (like in the LMA) the highest propagation speed is only one cell per update cycle. Which is to slow for, e.g., a realistic simulation of a collision of to large molecules.

3.7.4 Self-Replicating Cell

Ono and Ikegami [97] have constructed a simple model of primitive cells that simulates the dynamics of a set of symbolic chemicals on a two-dimensional grid. One of the substances autocatalytically replicates and produces membrane molecules by converting a kind of high-energy "substrate". This AC is able to organize itself into a cell-like structure that is dynamically stable (Fig. 8). They have shown that these cells can divide themselves into sub-cells spontaneously.

[13] Measured by the probability to find a molecule close to another

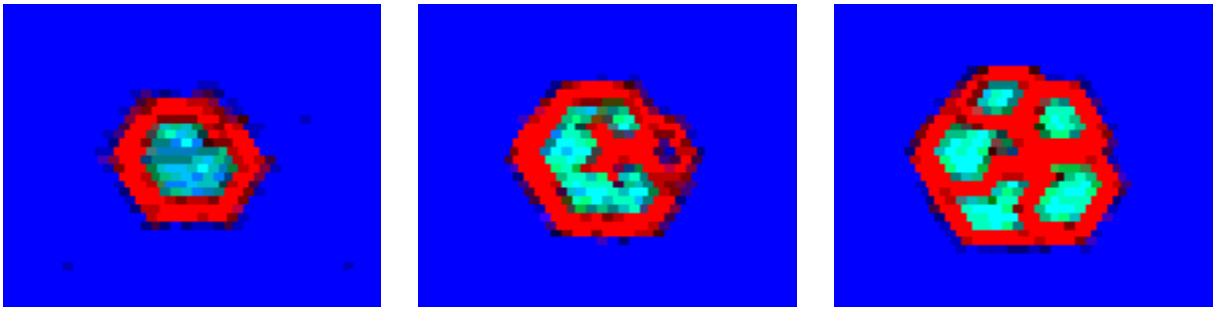


Figure 8: *Cell growth and cell division of a self-replicating cell by Ono and Ikegami [97].*

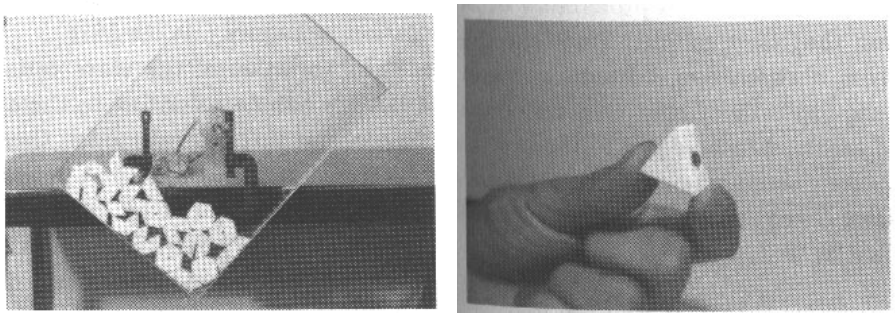


Figure 9: *A mechanical self-assembly system by Hosokawa et al. (from [64])*

3.8 Other Approaches

This section describes artificial chemistry approaches which can not easily be associated to one of the sections above. Nevertheless they represent very interesting and important variants demonstrating qualitatively additional mechanisms and properties.

3.8.1 Mechanical Artificial Chemistry

Hosokawa et al. [64] show that a mechanical self-assembly system can be regarded as a chemical system. This allows to build a model for the yield of the desired target structure by means of chemical kinetics theory. In [64] this approach is demonstrated by deriving an analytical model for a simple mechanical self-assembling system shown in Fig. 9. The basic units of the mechanical systems are regular triangular units which may form bonds by permanent magnets. This work demonstrates that artificial chemistry research is not limited to formal or algorithmic systems. Begin

Another physical model of a chemical system has been developed by Grzybowski, Stone, and Whitesides [55]. It consists of millimeter-size discs rotating at a liquid-air interface. An important difference to the work of Hosokawa et al. [64] is that the emergent structures are dissipative. This means, that the structures are only present as long as they are supplied with energy. The energy is fed into the system by a permanent magnet rotating underneath the liquid. End

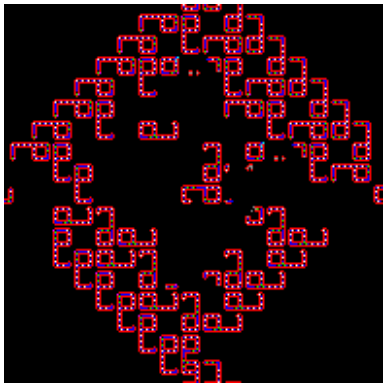


Figure 10: *Example of a set of self-replicating loops in a finite space by Sayama [111]*

3.8.2 The Chemical Metaphor in Cellular Automata

In Sec. 3.7 we have already seen that cellular automata can be used as a medium to simulate polymer chemistries. In this section we are going to discuss two more abstract models. They are more abstract in the sense that the definition of a molecule or reaction is implicit and depends on an observer or on an interpretation. In the lattice molecular systems described above it is clear what a molecule or a reaction is, because they are part of the model description. In the following approaches the model specification does not contain any notion of a molecule or reaction. They emerge as structures which are interpreted as chemical units.

Self-replicating Loops In cellular automata self-replicating patterns can appear. A famous example is the original work by von Neumann who defined a complex CA in order to demonstrate that self-replication is possible inside machines. In subsequent work by others simpler and more elegant cellular automata have been developed displaying self-replicating patterns, like Langton's self-reproducing loop [83]. A cellular automaton like this can be interpreted as an artificial chemistry where the molecules are the self-replicating structures. This becomes for example noticeable in the work by Sayama [111] who introduces structural dissolution in Langton's self-reproducing loop. In this work the loops begin to dissolve after replication (Fig. 10). Thus they have to permanently keep reproducing in order to keep the number of loops at a certain concentration level. Sayama also showed that in a bounded finite space competition occurs which leads to evolutionary effects. Furthermore interaction among loops take place which can be interpreted as a chemical reaction.

Embedded Particles in Cellular Automata as Molecules A different way to relate CAs to chemistry is to interpret a moving boundary between two homogeneous domains in a CA as a molecule [63]. Figure 11 shows an example of a one-dimensional CA. The figure shows the trace of the state-transition where two "particles" collide. The major difference to the previous approach is that particles become visible as space-time structures and that they are defined as boundaries and not as a connected set of cells with specific states.

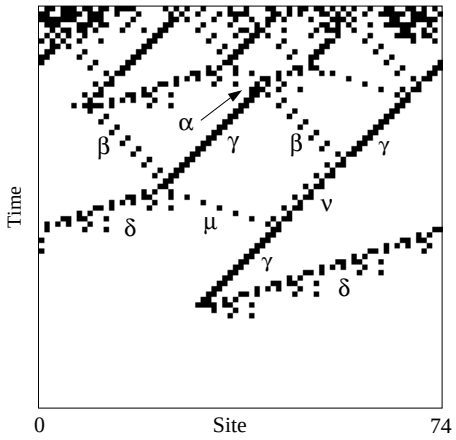


Figure 11: *Filtered version of a space-time diagram of a one-dimensional, binary-state, $r = 3$ CA. Homogenous domains are mapped to white, domain borders to black. Domain borders can be interpreted as particles or molecules. Figure taken from [63]*

3.8.3 Typogenetics

Typogenetics has been introduced 1979 by Hofstadter in order to illustrate the "formal logic of life" [62]. It can be regarded as one of the first artificial chemistries and has been further investigated by Morris, Varetto and others [95, 132, 133]. The molecules of the system are character sequences (called strands) over the alphabet `A, C, G, T`. The reaction rules are "typographic" manipulations based on a set of predefined basic operations such as cutting, insertion, or deletion of characters. A sequence of such operations form a unit (called enzyme) which may operate on a character sequence like a Turing machine on its tape (Fig. 12).

A strand codes for several enzymes which are separated by the "duplet" `AA`. To translate a character sequence into a sequence of operations a pair of two characters codes for one operation. For example `AC` codes for "cut" which cuts a strand at the current binding position of the enzyme (see Fig. 12). The initial binding position of an enzyme depends in a non-trivial way on its coding sequence. In Fig. 12 the first enzyme binds initially to `G`. If the strand contains more than one character of this kind, one will be randomly chosen.

Double stranded molecules can be formed, too. There are two operations which create double stranded molecules: "copy mode on" and "copy mode off". If the copy mode is active, movement of the enzyme creates an inverse copy of characters it is touching. The enzyme may also switch from one strand to its parallel strand by a "switch" operation (Fig. 12). Typogenetic is very complex because the size of molecules is variable, a reaction may produce a variable number of products and the reaction mechanism is non-deterministic.

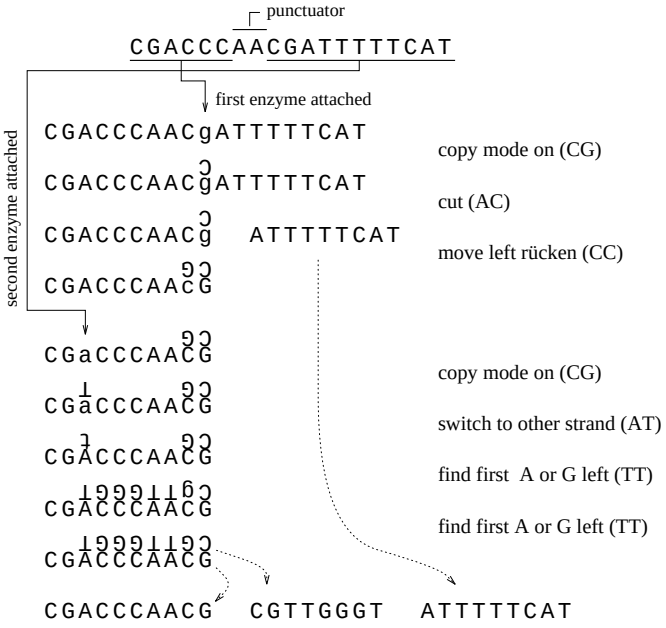


Figure 12: *Example of a reaction in a Typogenetics model. Two enzymes coded by the strand* `CGA....AT` *operate on a copy of the same strand. The result are three reaction products. The current binding position of an enzyme is depicted by a lower case character. Example taken from [95].*

4 Applications

As has been argued above, the applications of artificial chemistries can be subdivided into three domains: (1) modeling, (2) information processing, and (3) optimization. By "application" we refer not only to industrial applications but also to applications where the chemical metaphor is used to build a model of a system in order to extend the knowledge about our environment.

4.1 Modeling

Most of the AC work falls into the domain of modeling. Within this domain mainly bio-chemical systems are modeled. Exceptions are, for instance, the Chemical Abstract Machine which models concurrent computer processes [20] (Sec. 3.1.1) or the Random Prolog Processor applied to model social systems [122, 123, 125].

The chemical metaphor can also serve to model higher structures such as population dynamics, ecological systems, social systems, or economic markets. The similarity of these systems and their relation to the chemical metaphor becomes particularly clear in the replicator equation [128, 112]

$$\frac{dx_i}{dt} = x_i(f_i(x_1, \ldots, x_n) - \Phi(x_1, \ldots, x_n)) \tag{23}$$

which describes the dynamics of n replicating types. The function f_i can be interpreted as the catalytic influence of all other types on the replication rate of x_i or as the fitness of type i which may depend on the concentration of all other types. The dilution flux Φ is given by Begin

$$\Phi(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i f_i(x_1, \ldots, x_n) \tag{24}$$

and can also be interpreted as the average fitness ([61], p. 67). The dilution flux ensures End that the state of the system stays inside the concentration simplex S_n. On molecular level the replicator equation is applied to model ensembles of replicating molecules (e.g., polymers). In this category falls the hypercycle [47] which dynamics can be described by a special form of the replicator equation. On an ecological scale the replicator equation describes the dynamics of interacting populations. In this case replication rate constants can be associated with fitness values of certain groups or species. The replicator equation is equivalent to the Lotka-Voltera equation [59] which demonstrates the close relation between chemical and ecological dynamics. Even for social and economic systems the replicator equation can serve as a model [60].

4.1.1 Modeling (Bio-)Chemical Systems

Artificial chemistries modeling (bio-)chemical systems can be subdivided according to their level of abstraction (Sec. 2.3.4). ACs like the polymer automata [90] or the Belusov-Zhabotinsky-reaction formulated in ARMS [120, 121] model concrete chemical systems

in an analogous way (Sec. 2.3.4). Molecule and reaction in the model corresponds to molecule and reactionin reality, respectively.

Approaches like AlChemy, tape-rewriting systems, assembler automata, number chemistries or random chemistries (e.g., GARD or random ARMS) model the dynamics of chemical-like organizations in a more abstract manner. These are the approaches to which the term "artificial" refers. Strictly speaking we should have these abstract models of chemistry in mind when using the term "artificial chemistry" in a modeling context. If we encompass all kinds of chemical models the term "artificial chemistry" would lose some substance. Of course the boundary line is fuzzy.

4.1.2 Evolution and Self-assembly

Begin

Two important ways to generate biochemical complexity are (1) by means of self-assembly and (2) by means of evolution. We can use these two aspects as another way to distinguish between different AC approaches.

Self-assembly produces a more complicated structure by local interaction of independent entities without central control. Typical examples for self-assembly are the formation of micells or the morphogenesis of phages. It is believed that the transition from a non-self-maintaining population of molecules to a self-maintaining population, which later developed into a population of aggregates that we would call living, has occurred through a process of chemical evolution and self-assembly [31]. Self-assembly not only plays an important role in the origin of living systems, but also in their operation today. Furthermore, self-assembly can be employed as a strategy to build nano-scale structures [134], such as thin films or molecular wires, or to solve computational problems [88].

When self-assembly is simulated we have to include a model of a (physical) space wherein the entities are located. It is typically for such simulation models that the space of interaction is the same as the space of movement. Thus, we cannot easily (or naturally) obtain a model in terms of the (S, R, A) scheme by separating the reaction rules R from the algorithm A that describes the reaction space. Furthermore, interactions may depend on the movement (e.g., velocity) or relative orientation of the interacting entities in space [135].

There is a large body of work in the domain of computational chemistry that uses Monte Carlo methods in order to simulate self-assembly processes accurately, e.g., [38]. These approaches aim at realistic models for direct comparison of simulation results with experimental real-world data. On the other hand, artificial chemistry approaches do not emphasize comparability on the microscopic level, rather they aim at revealing the "logic" of certain phenomena, such as the emergence of dynamical hierarchies [89], the formation of micells [37, 45, 113] or autopoietic cells [131, 93, 97], or the spontaneous formation of aggregates which undergo division and evolution [135] (Sec. 3.7). It might be added that physical models are also nicely suited to investigate self-assembly [55, 64] (Sec. 3.8.1).

The AC literature on evolution is much larger than that on self-assembly (see Sec. 3). AC work on evolution can be sub-classified according to whether it concentrates on the origin (of evolution) [9, 40, 51, 66, 91, 117], or whether they assume prerequisites, such as replication or individuals which differ in fitness, as given predefined components of the model [47, 69, 112]. Up to now, most AC publications on evolution do not include

self-assembly. It would be interesting for the future to join both principles in order to investigate their mutual influence. End

4.1.3 Ecological Modeling with Artifical Chemistries

AC as a Sub-System of an Ecological Model In an ecological model an artificial chemistry can provide a fundamental "currency" if mass conservation and energy conservation is respected. In such ecological models organisms may absorb chemicals from their environment and metabolize them [44, 43]. The organisms may even consist of chemicals which are released after death [26].

An example for this approach is the metabolically-based artificial ecosystem model EVOLVE IV [26, 27] which is an advancement of a series of evolutionary ecosystem computer models originated in 1969 [33, 108] The objective of these models is to investigate the conditions under which an evolutionary process comparable to that observed in nature occurs [26]. Species formation is one of the investigated phenomena, e.g., the development of diverse quasispecies [33, 35]. The aim of EVOLVE IV is (a) to generate a niche structure reminiscent of the niche structure of a natural ecosystem, (b) to investigate the conditions under which niche diversification can occur. Brewster and Conrad [26] describe experiments where the population is initialized with hand-written organisms. They consist of 22 essential genes (e.g., sugar decomposition) and one non-essential gene (conjugation gene). There are two complementary genes (photosynthesis and scavenging gene) so that the population may diversify by specializing on one of these genes. At least one of these two genes is required for survival. In simulations most organisms carried both genes (if initially supplied with both) ([26], p. 480).

AC as an Ecological Model Suzuki, Takabayashi and Tanaka [119] applied the abstract chemical re-writing system (ARMS) to model an ecological system in which plants respond to herbivore feeding activity by producing volatiles that in turn attract carnivorous natural enemies of the herbivores. Such defense mechanisms have been reported in several tritrophic systems. The volatiles are not the mere result of mechanical damage, but are produced by the plant as a specific response to herbivore damage. Suzuki et al. compared the case where plants produce herbivore-induced volatiles vs the case where they do not. They found that there was a case where herbivore-induced volatiles that attract carnivores resulted in the population increase of the herbivores.

4.1.4 Social Modeling with Artificial Chemistries

Computer simulations are a small subfield in theoretical or mathematical sociology but the number of contributions is constantly growing. In the following section a concrete example is presented where a constructive artificial chemistry is applied to model social dynamics [122]. The model is based on the Random Prolog Processor (RPP) [125] which is a non-deterministic model for evaluation and motivated by works of Berry/Boudol, Fontana, Benatre, and others.

Molecules: There are three types of molecules (called information molecules): Facts, rules, and goals. They are statements of a logical programming language.

Reactions: Szuba and Stras [125] report on four implemented general collision rules (inferring diagrams). What kind of inferring diagram applies depends on the type of the colliding molecules.

Dynamics: The molecular dynamics is simulated as an explicit Brownian motion in a 3-D Euclidian space. Szuba and Stras [125] simulated 200-300 molecules, which required a parallel 8 processor SGI (1997). A simulation cycle consists of (1) computation of random displacement vectors such that boundary conditions are not violated (e.g., city border, reaction vessel border), (2) movement of molecules, (3) generation of subsets of molecules which are in a rendezvous distance d, (4) processing of reactions (generate logic formulas, evaluate them, create new molecules if clauses are not false,...)

Szuba and Stras [124] applied the RPP to model social systems. In their model a social system consists of facts, rules, and goals called molecules here. A membrane concept similar to the membrane concept of the CHAM allows a hierarchical structure. Because the computational resources needed for simulating hierarchical systems are very high only non-hierarchical systems are simulated in [124]. The motivation is to create measures and tools to evaluate the inference power of human social structures (e.g., cities or nations).

As a benchmark problem Szuba and Stras introduced the *N-step inference*. The N-step inference problem consists of N rules, one fact and one goal. To infer the goal the N rules have to be applied sequentially in a specific order. Szuba and Stras showed that a system which contains a "city" is able to solve the problem much faster than a system without a city. In order to model a city a spherical subspace of the RPP reactor influences the movement of the molecules in the following way: (1) Molecules that enter the "city" can leave it only after they stayed a certain amount of time in the "city". (2) Molecules move slower inside the city.

The obtained results are from a social science and dynamical system point of view not surprising. But the introduced methods are fascinating and a promising approach to build individual based social simulation models. It should also be stressed that the underlying reaction mechanism based on logical inference is quite general and allows to formulate a huge variety of interaction models easily.

4.2 Information processing

Another major application of AC is the field of information processing. Since every living entity can be seen formally as an information processing system, generating output (in general a certain behavior) as a result of processing the input (the current internal state and the environmental inflow), it appears evident that the basic mechanism of information processing is chemistry, and thus, to study artificial information processing, AC is an appropriate tool.

Molecules: The idea becomes clear if we have the following metaphor in mind: as was stated by us [11], the data to be processed (input and output) shall be seen as molecules, carrying a certain 'meaning'. In this context, the general term information is reduced to *pragmatic information*.

Reactions: The processing of data can now be regarded as the molecule-molecule interactions (reactions), in which each of the residing 'meanings' is either propagated,

multiplicated, eliminated, or processed.

Dynamics: To make the processing of information by means of reactions a reasonable approach, the dynamics of the AC is a crucial part of the system.

The research of information processing with AC divides into two different approaches, that will be explained in some detail in the next two sections.

4.2.1 Artificial Chemical Computing

Artificial Chemical Computing (ACC) deals with information processing systems consisting of an artificial chemistry as the main processing entity. This processing of information– common to all approaches–is then used for different reasons:

First, and this is the main approach, ACC is used for control tasks, especially for mobile robots. Husbands [65] used an AC to get better results with an artificial neural network for robot control. In his model neurons are able to emit substances which diffuse and modulate transmission functions of other neurons. Brooks [29] used a hormone system to achieve asynchronous information flow and coherent behavior in a distributed parallel control architecture for a humanoid robot torso. The hormone system is equivalent to an artificial chemistry without reactions, so that hormones just carry information but do not process them. Adamatzki et al. [1, 2] and we [139] used excitable lattices and simplified enzyme-substrate kinetics, respectively, to control real mobile robots.

Lugowski, Shackleton, Aoki [6, 87, 115], to give some approaches, intended to establish a new sight on a parallel distributed computer architecture, completely different from the well known and common von-Neumann-architecture, Conrads and Zauner [34, 36, 137, 136] started to implement and analyze a simulator for shape-based or pattern-recognition-based biochemical motivated complex reaction networks. Astor [8] used AC to control the growth of artifical neural networks, one of us [12] used the processing capabilities to implement a "molecular" solution to the traveling salesman problem.

Due to the complexity of AC, all approaches used more or less manually designed instances of AC to get the desired results. It is up to now an unsolved problem to automatically generate ACs with properties suitable for the given problem. One possible solution could be the evolution of reaction networks [86, 140], that in turn would result in "emergent programming" of AC.

4.2.2 Real Chemical Computing

In the field of Real Chemical Computing (RCC), the well established DNA-Computing plays a dominant role. There are, however, several other approaches which establish a research area that deals with the idea of using real chemistry to perform computations. Hjemfeld et al. [58], for instance, investigate the possibilities of realizing artificial neural networks and Turing machines in vitro. Arkin [7] discovered computational functions in biochemical reaction networks and therewith proved Hjemfeld's thesis. Adamatzki et al. proposed the use of oscillating reactions like the Belousov-Zhabotinsky reaction to control mobile robots [1].

The most promising bio-molecule for a technical application which would be able to compete with classical electrical computers is *bacteriorhodopsin*. Bacteriorhodopsin is a

molecule found in the membrane of *Halobacterium halobium*. The molecule is responsible for the conversion of solar light into chemical energy. In the mid 80s it has been found that it can be used as a holographic recording medium. Important properties of this molecule are that its confirmational state can be changed by photons of specific wave lengths and that it is very robust and not destroyed when stimulated many times. In addition the current conformational state can be easily detected by using light because the adsorbtion spectra of bacteriorhodopsin depends on its conformational state [18]. Potential applications of bacteriorhodopsin are high-density memories [23], hybrid electro-optical neural networks [57], and optical pattern recognition [56].

DNA Computing Since DNA Computing is now a well established special branch of RCC, we will not discuss it in much detail here but only give a short introduction.

DNA Computing (DNA-C) consists of two branches: theoretical DNA-C investigates the possibilities of DNA with regard to well-known problems of computer science (e.g., universal computing [19, 25], cryptology [24], or acceleration of computation [85]). A DNA computer model can be regarded as a constructive, analogous artificial chemistry. It is analogous because it should model a real bio-chemical system accurately such that a molecule in simulation can be identified with a real molecule and a reaction going on in simulation should also have its counterpart in reality. One expects results forecasting the computational power of DNA.

On the other hand, DNA-C established as "wet computing". This is mainly due to Adleman's famous experiments [5], where he solved a variant of the Hamiltonian path problem (a NP-complete problem in computer science) with real DNA. In the meantime these experiments have been reproduced and others are conducted that investigate the computational power and the applicability ofthis approach under real conditions.

4.3 Optimization

Another application of AC is the field of optimization. There are evolutionary algorithms (EA), well established since the late sixties, which benefit from the possibility to control evolution with externally defined selection scheme and fitness function. Because of the ability of ACs to create evolutionary behavior (or even self-evolution [40]), it is not a big step towards using AC for evolutionary optimization. Koza [77] created self-replicating computer programs which are self-improving and evolving. This system can be regarded as an artificial chemistry where the molecules are computer programs and reactions take place while programs are executed.

Kanada and Hirokawa [71, 70] (Sec. 3.1.3) have demonstrated successfully the application of the Chemical Casting Model (CCM) as a new method for optimization to different constraint satisfaction problems such as graph coloring, the N-queens problem, and the Traveling Salesman Problem (TSP). Molecules (atoms) and reaction rules depend on the type of the search problem. In general, atoms are components of a potential solution of the problem. Another molecular solution of the TSP was suggested by one of us [12]. Ray [106] discussed the possibility of optimization in the context of Tierra.

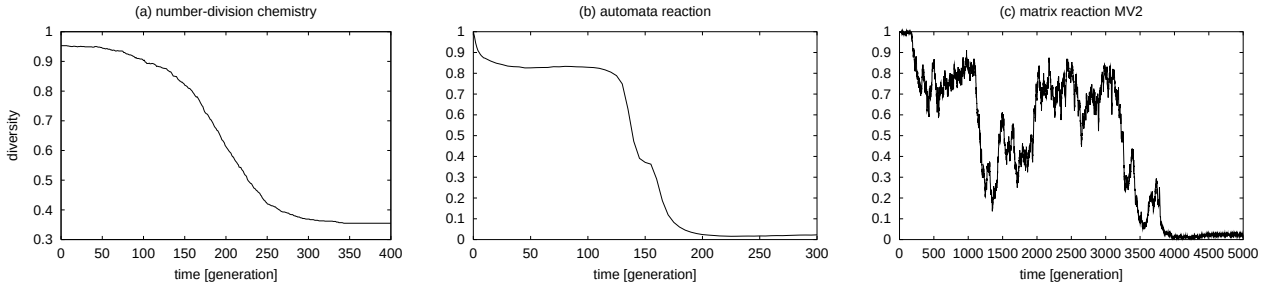


Figure 13: *Examples of diversity reduction which appears in different artificial chemistries: (a) number-division chemistry (prime number chemistry), (b) automata reaction, (c) variable length matrix reaction (MV2).*

5 Common Phenomena

Despite of the vast diversity of the different approaches in the field of artificial chemistry research, common phenomena can be observed.

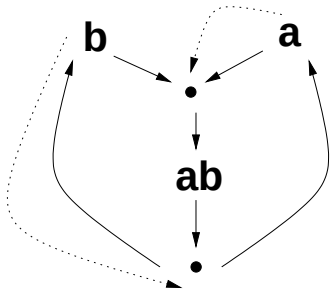
Reduction of Diversity There is a force towards reduction of diversity. This becomes visible in experiments where the reactor is initialized with maximum diversity as shown for example in Fig. 13. In nearly all artificial chemistries reported in the literature the diversity reduces to either (1) a single self-replicating molecule, (2) an inert population in which no reaction takes place, (3) a simple network consisting of a few interacting molecules, or (4) a complex reaction network which may change over time and which is not limited to a finite set of elements.

There is of course also a force increasing the diversity in constructive artificial chemistries. A typical example where this force becomes visible are variable length polymer chemistries where reactions may produce longer and longer sequences thus increasing the diversity.

Formation of Densely Coupled Stable Networks From a randomly initialized population the emergence of closely coupled stable reaction networks can often be observed. This phenomenon appears in various systems despite of the different representation of molecules and reaction rules. Examples are: Level-0 organizations in the lambda-chemistry, matrix-reaction, automata-reaction, Turing-machines on tape, catalytic polymer networks, assembler automata (e.g., Tierra), and others. Figure 14 shows three examples of stable reaction networks from different ACs.

Syntactic and Semantic Closure In strongly constructive artificial chemistries the molecules of the above mentioned stable networks (organizations) show similarities in structure and function. This phenomenon has been analyzed by Fontana and Buss in detail and called syntactic and semantic closure, respectively [50, 52, 49]. Figure 15 demonstrates that this phenomenon also appears in totally different artificial chemistries such as matrix-multiplication chemistry or abstract automata chemistries. Following Fontana one

No	string	0	1	2	3
0	7240a7ef	2	1	2	1
1	7240a7ea	0	0	0	0
2	7240a7eb	3	0	3	0
3	7240a7ee	1	1	1	1



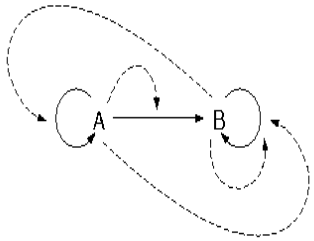


Figure 14: **Left:** *Stable network of binary strings with reaction matrix.* **Middle:** *Network of polymer reactions. Dotted lines indicate catalytic activity.* **Right:** *Simplest stable network of λ-expressions.* $A = \lambda s_1 \cdot (s_1)\lambda s_2 \cdot \lambda s_3 \cdot (s_3)\lambda s_4 \cdot \lambda s_5 \cdot (s_5)s_4$, $B = \lambda s_1 \cdot (s_1)\lambda s_2 \cdot \lambda s_3 \cdot (s_3)s_2$.

can derive a grammar which fully describes the subset of molecules forming the surviving organization. Based on this abstract syntactic description one can also define an algebra which specifies the interactions among the molecules (semantic closure). This semantic description abstracts from the underlying reaction mechanism. Figure 15 shows examples of syntactical similar molecular organizations taken from different artificial chemistries. The similar syntactical structure allows to describe the structure of the dominating ensemble of species in a more abstract way by a grammar and their relations by an algebra. By doing so, an organization can be described by a grammar (specifying the syntax) and an algebra (specifying the semantics) in an abstract way, so that the description is independent from the underlying reaction level. Figure 16 shows a simple example of a level-0 organization taken from an automata chemistry [40].

λ-chemistry	automata reaction	matrix reaction
$\lambda x_1.\lambda x_2.\lambda x_2$	7240a7ef	110000
$\lambda x_1.\lambda x_2.\lambda x_3\lambda x_2$	7240a7ea	111000
$\lambda x_1.\lambda x_2.\lambda x_3.\lambda x_3$	7240a7eb	11110000
$\lambda x_1.\lambda x_2.\lambda x_3.\lambda x_4.\lambda x_3$	7240a7ee	11100000

Figure 15: *Example of the syntactical similarity. Shown are samples from a λ chemistry [50], an automata chemistry [40] and a chemistry based on matrix multiplication.*

Evolution and Punctuated Equilibrium Reaction networks may also evolve over time as has been shown for the autocatalytic polymer chemistry by Bagley et al. [10]. Evolution is usually induced by external noise[14] causing variations in the population. Bagley et al. for example introduce randomly generated catalytic links between molecular sequences which may lead to a restructuring of the metabolic network. Another frequently used method is to introduce random point mutations (e.g., bit-flips) in the structure of molecules. Examples are: machine-tape chemistries [66, 67, 91, 129], assembler automata like Tierra [107], Avida [3], or Coreworld [103].

[14]Called passive mutation by Ikegami and Hashimoto [66, 67]

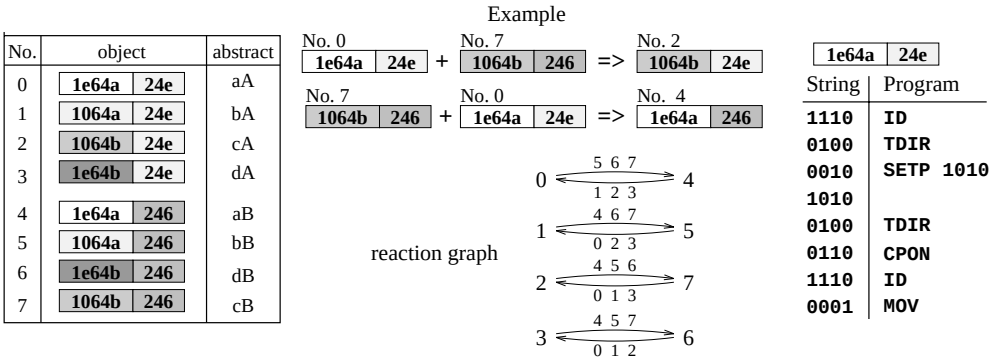


Figure 16: *Example of sematic closure of a level-0 organization. Example taken from the automata reaction [40]. The self-maintaining organization consists of 8 molecular types. Their syntactic similarity allows to represent them in the form xY where $x \in \{a, b, c, d\}$ and $Y \in \{A, B\}$. The semantic closure allows now to describe the interactions valid for this organization by a simple algebraic rule: $xY + x'Y' \longrightarrow xY + x'Y' + x'Y$.*

Evolution may also take place without external, explicit mutations. In this case variation is only performed by the molecules themselves. This type of variation is called active mutation by Ikegami and Hashimoto. It has been demonstrated that an artificial chemistry with only active mutation can show evolutionary phenomena either in presence of an explicit, externally defined fitness function [77] or even without any explicit fitness function [40, 42].

6 Discussion and Outlook

This paper gave a survey of the present state of research in the field of artificial chemistries. It provided an overview about diverse approaches presently discussed in the literature. The goal of our paper was to introduce this diversity using a consistent nomenclature for specific aspects of ACs and a systematic description for a meaningful comparison between different approaches. After basic concepts the main research areas were introduced, followed by different fields of applications, such as modeling, information processing and optimization. Common features of ACs, e.g., reduction of diversity of molecules or the formation of densely coupled stable networks, were discussed in Chapter 5.

We would like to argue that the knowledge accumulated in studying artificial chemistries will provide a fertile ground for new ideas about the origin of life and for prebiotic evolution. By using the freedom of formulation that is offered by abstract systems resembling chemistry, the problem of life's self-organization might be better explored than by sticking to the one real chemistry that is available in Nature. This does not mean, however, that one should detach from the natural system altogether. If, for instance, generic phenomena of ACs can be connected successfully to natural phenomena, these phenomena might become qualitatively predictable. Begin

Important questions remain though: What level of abstraction for an AC is appropriate to be useful in the context of life's origin? Which key ingredients are missing in current AC systems? Do we have to incorporate detailed physical/chemical knowledge? What does evolution mean in the context of a (artificial) chemical system? What role does self-assembly play in the emergence of novel functionality? What is the relation between evolution and self-assembly? Does "information processing" emerge by way of evolution? And if so, how? End

If ACs are a subject of study in their own right, a number of questions remain to be answered: What are "natural laws" of artificial chemistries? What are organizations and how can an AC system self-organize? How can an artificial chemistry be investigated and be analyzed? Is an artificial chemistry able to create information? How can "evolution" be measured in general? How many meta-levels of evolution are there?

As it looks now, ACs will have many different applications. The ubiquity of interaction among isolated objects in our world stands testimony to this potential. Starting from an analysis of possible interactions, these should be formulated in terms of objects and their interaction features. Dynamical processes will be generated from these, and can be set up in a way useful for the purpose in mind. So it might be asked, what is the relation of ACs to other domains of research, such as population dynamics, immune system dynamics, social dynamics, or economic system dynamics?

It is our conviction that artificial chemistries in their most general formulation as abstract systems of objects which follow arbitrary rules of interaction in combinatorial spaces will in due time contribute to many different fields of inquiry. Therefore we would like to encourage readers to study systems of their own interest from this perspective.

7 Acknowledgements

This research has been supported by the DFG (*Deutsche Forschungsgemeinschaft*) under grant Ba 1042-2. We thank Chris Adami, Jens Busch, John McCaskill, Pietro Speroni di Fenizio, Yasuhiro Suzuki, and Klaus-Peter Zauner for fruitful discussions and comments on topics of this paper. We especially thank Steen Rasmussen and the anonymous reviewer for their detailed critique which has led to a significant improvement of the manuscript. We additionally thank Andre Skusa and Christian Düntgen for technical assistance. This work was originally inspired by the workshop "The right stuff" of the ALIFE VI conference, where several of the challenging questions were raised.

References

[1] A. Adamatzky, O. Holland, N. Rambidi, and A. Winfield Wet artificial brains: Towards the chemical control of robot motion by reaction-diffusion and exitable media. In Dario Floreano, Jean-Daniel Nicoud, and Francesco Mondada, editors, *Proc. Fifth European Conference on Artficial Life*, pages 304–13, Berlin, 1999. Springer.

[2] Andrew Adamatzky and Owen Holland. Edges and computation in excitable media. In Christoph Adami, Richard K. Belew, Hiroaki Kitano, and Charles Taylor, editors, *Artificial Life VI*, pages 379–383, Cambridge, MA, USA, June 27–29 1998. MIT Press.

[3] C. Adami and C. T. Brown. Evolutionary learning in the 2D artificial life system avida. In R. A. Brooks and P. Maes, editors, *Artificial Life IV*, pages 377–381, Cambridge, MA, 1994. MIT Press.

[4] Christoph Adami. *Introduction to Artificial Life*. Springer, New York, NY, 1998.

[5] L. M. Adleman Molecular computation of solutions to combinatorical problems. *Science*, 266:1021, 1994.

[6] T. Aoki, M. Kameyama, and T. Higuchi. Interconnection-free biomolecular computing. *Computer*, 25:41–50, 1992.

[7] A. Arkin and J. Ross Computational functions in biochemical reaction networks. *Biophysical Journal*, 67:560–578, 1994.

[8] J. C. Astor and C. Adami. A developmental model for the evolution of artificial neural networks. *Artif. Life*, 6(3):189–218, 2000.

[9] Richard J. Bagley and J. Doyne Farmer. Spontaneous emergence of a metabolism. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 93–140, Redwood City, CA, 1992. Addison-Wesley.

[10] Richard J. Bagley, J. Doyne Farmer, and Walter Fontana. Evolution of a metabolism. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 141–158, Redwood City, CA, 1992. Addison-Wesley.

[11] W. Banzhaf, P. Dittrich, and H. Rauhe. Emergent computation by catalytic reactions. *Nanotechnology*, 7(1):307–314, 1996.

[12] Wolfgang Banzhaf. The "molecular" traveling salesman. *Biol. Cybern.*, 64:7–14, 1990.

[13] Wolfgang Banzhaf. Self-replicating sequences of binary numbers – foundations I and II: General and strings of length n = 4. *Biol. Cybern.*, 69:269–281, 1993.

[14] Wolfgang Banzhaf. Self-organization in a system of binary strings. In R.A. Brooks and P. Maes, editors, *Artificial Life IV*, pages 109–119, Cambridge, MA, 1994. MIT Press.

[15] Wolfgang Banzhaf. Self-replicating sequences of binary numbers: The build-up of complexity. *Complex Syst.*, 8:215–225, 1994.

[16] Wolfgang Banzhaf. Self-organizing algorithms derived from RNA interactions. In Wolfgang Banzhaf and Frank H. Eeckman, editors, *Evolution and Biocomputing*, volume 899 of *LNCS*, pages 69–103. Springer, Berlin, 1995.

[17] Wolfgang Banzhaf, Peter Dittrich, and Burkhard Eller. Selforganization in a system of binary strings with topological interactions. *Physica D*, 125:85–104, 1999.

[18] V. Y. Bazhenov, M. S. Soskin, V. B. Taranenko, and M. V. Vasnetsov Biopolymers for real-time optical processing. In H. H. Arsenault, editor, *Optical Processing and Computing*, pages 103–44, San Diego, 1989. Academic Press.

[19] D. Beaver. A universal molecular computer. Technical report CSE-95-001, Penn State University, 1995.

[20] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96(1):217–248, 1992.

[21] Hugues Bersini. Design patterns for an object oriented computational chemistry. In Dario Floreano, Jean-Daniel Nicoud, and Francesco Mondana, editors, *Proc. Fifth European Conference on Artificial Life (ECAL'99)*, pages 389–298, Berlin, 1999. Springer.

[22] Hugues Bersini. Reaction mechanisms in the oo chemistry. In M. A. Bedau, J. S. McCaskill, N. H. Packard, and Steen Rasmussen, editors, *Artificial Life VII*, pages 39–48, Cambridge, MA, 2000. MIT Press.

[23] R. R. Birge Protein-based optical computing and memories. *Computer*, 25:56–67, 1992.

[24] D. Boneh, C. Dunworth, and R. J. Lipton. Breaking DES using a molecular computer. Technical Report CS-TR-489-95, Princton University, 1995.

[25] D. Boneh, C. Dunworth, R. J. Lipton, and J. Sgall. On the computational power of dna. *Discret Appl. Math.*, 71(1-3):79–94, 1996.

[26] J. J. Brewster and M. Conrad. EVOLVE IV: a metabolically-based artificial ecosystem model. In V.W. Porto, N. Saravannan, D. Waagen, and A.E. Eiben, editors, *Evolutionary Programming VII*, pages 473–82, Berlin, 1998. Springer.

[27] J. J. Brewster and M. Conrad. Computer experiments on the development of niche specialization in an artificial ecosystem. In *Proc. 1999 Congress on Evolutionary Computation (CEC'99)*, pages 439–44. Piscataway: IEEE, 1999.

[28] J. Breyer, J. Ackermann, and J. McCaskill Evolving reaction-diffusion ecosystems with self-assembling structure in thin films. *Artificial Life*, 4(1):25–40, 1999.

[29] Rodney A. Brooks Coherent behavior from many adaptive processes. In Dave Cliff, Philip Husbands, Jean-Arcady Meyer, and Steward Wilson, editors, *From animals to animats 3*, pages 22–29, Cambridge, MA, 1994. MIT Press.

[30] Cristian Calude and Gheorghe Păun. *Computing with Cells and Atoms*. Taylor and Francis, 2000.

[31] Melvin Calvin. *Chemical Evolution: Molecular Evolution Towards the Origin of Living Systems on the Earth and Elsewhere*. Oxford University Press, New York, 1969.

[32] T. Clark. *A Handbook of Computational Chemistry, A Practical Guide to Chemical Structure and Energy Calculations*. Wiley, New York, 1985.

[33] M. Conrad. *Computer Experiments on the Evolution of Co-adaptation in a Primitive Ecosystem*. PhD thesis, Stanford University, 1969.

[34] M. Conrad. Molecular computing: The key-lock paradigm. *Computer*, 25:11–22, 1992.

[35] M. Conrad and M. M. Pattee Evolution experiments with an artificial ecosystem. *J. theort. Biol.*, 28:293–409, 1970.

[36] M. Conrad and K.-P. Zauner. Conformation-driven computing: A comparison of designs based on DNA, RNA, and protein. *Supramolecular Science*, 5(5-6):787–790, 1998.

[37] P. V. Coveney and J. A.D Wattis. Becker-doring model of self-reproducing vesicles. *J. Chem. Soc.-Faraday Trans.*, 94(2):233–246, 1998.

[38] E. J. Dawnkaski, D. Srivastava, and B. J. Garrison. Growth of diamond films on a diamond 001(2x1):h surface by time dependent monte carlo simulations. *J. Chem. Phys.*, 104(15):5997–6008, 1996.

[39] A. K. Dewdney In the game called core war hostile programs engage in a battle of bits. *Sci. Amer.*, 250:14–22, 1984.

[40] P. Dittrich and W. Banzhaf. Self-evolution in a constructive binary string system. *Artificial Life*, 4(2):203–220, 1998.

[41] Peter Dittrich and Wolfgang Banzhaf. A topological structure based on hashing - emergence of a "spatial" oranization. In *Fourth European Conference on Artificial Life (ECAL'97)*, Brighton, UK, 28-31 July 2000, http://www.cogs.susx.ac.uk/ecal97/, 1997.

[42] Peter Dittrich, Jens Ziegler, and Wolfgang Banzhaf. Mesoscopic analysis of self-evolution in an artificial chemistry. In C. Adami, R. Belew, H. Kitano, and C. Taylor, editors, *Artificial Life VI*, pages 95–103, Cambridge, MA, 1998. MIT Press.

[43] K. Downing and P. Zvirinsky. The simulated evolution of biochemical guilds: Reconciling gaia theory and natural selection. *Artif. Life*, 5(4):291–318, 1999.

[44] Keith Downing. Exploring gaia theory: Artificial life on a planetary scale. In M. A. Bedau, J. S. McCaskill, N. H. Packard, and Steen Rasmussen, editors, *Artificial Life VII*, pages 90–99, Cambridge, MA, 2000. MIT Press.

[45] L. Edwards and Y. Peng. Computational models for the formation of protocell structures. *Artif. Life*, 4(1):61–77, 1998.

[46] R. Ehricht, T. Ellinger, and J. S. McCascill. Cooperative amplification of templates by cross-hybridization (CATCH). *European Journal of Biochemistry*, 243(1/2):358–364, 1997.

[47] M. Eigen and P. Schuster. *The Hypercycle - A Principle of Natural Self-Organization.* Springer, Berlin, 1979.

[48] J. D. Farmer, S. A. Kauffman, and N. H. Packard Autocatalytic replication of polymers. *Physica D*, 22:50–67, 1986.

[49] W. Fontana. Algorithmic chemistry. In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, *Artificial Life II*, pages 159–210, Redwood City, CA, 1992. Addison-Wesley.

[50] W. Fontana and L. W. Buss 'The arrival of the fittest': Toward a theory of biological organization. *Bull. Math. Biol.*, 56:1–64, 1994.

[51] W. Fontana and L. W. Buss What would be conserved if 'the tape were played twice'? *Proc. Natl. Acad. Sci. USA*, 91:757–761, 1994.

[52] W. Fontana and L. W. Buss The barrier of objects: From dynamical systems to bounded organization. In J. Casti and A. Karlqvist, editors, *Boundaries and Barriers*, pages 56–116, Redwood City, MA, 1996. Addison-Wesley.

[53] W. Fontana, G. Wagner, and L. W. Buss Beyond digital naturalism. *Arficial Life*, 1/2:211–227, 1994.

[54] C. Furusawa and K. Kaneko. Emergence of multicellular organisms with dynamic differentiation and spatial pattern. *Artificial Life*, 4:79–93, 1998.

[55] B. A. Grzybowski, H. A. Stone, and G. M. Whitesides. Dynamic self-assembly of magnetized, millimetre-sized objects rotating at a liquid-air interface. *Nature*, 405(6790):1033–1036, 2000.

[56] N. Hampp, C. Bräuchle, and D. Oesterhelt Mutated bacteriorhodopsins: Competitive materials for optical information processing? *Materials Research Society Bulletin*, 17:56–60, 1992.

[57] D. Haronian and A. Lewis Elements of a unique bacteriorhodopsin neural network architecture. *Applied Optics*, 30:597–608, 1991.

[58] A. Hjelmfelt, E. D. Weinberger, and J. Ross Chemical implementation of neural networks and turing machines. *Proc. Natl. Acad. Sci. USA*, 88:10983–10987, 1991.

[59] J. Hofbauer On the occurence of limit cycles in the Volterra-Lotka equation. *Nonlinear Analysis*, 5:1003–7, 1981.

[60] J. Hofbauer and K. Sigmund. *Dynamical Systems and the Theory of Evolution*. University Press, Cambridge, UK, 1988.

[61] Josef Hofbauer and K. Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge University Press, Cambridge, UK, 1998.

[62] D. R. Hofstadter *Gödel, Escher, Bach: An Eternal Golden Braid*. Basic Books,Inc., New York, NY, 1979.

[63] W. Hordijk, J. P. Crutchfield, and M. Mitchell. Embedded-particle computation in evolved cellular automata. In T. Toffoli, M. Biafore, and J. Leäo, editors, *PhysComp96*, pages 153–8. New England Systems Institute, 1996.

[64] Kazuo Hosokawa, Isao Shimoyama, and Hirofumi Miura. Dynamics of self-assembling systems - analogy with chemical kinetics. In R.A. Brooks and P. Maes, editors, *Artificial Life IV*, pages 172–180, Cambridge, MA, 1994. MIT Press.

[65] Phil Husbands. Evolving robot behaviours with diffusing gas networks. In P. Husbands and J.-A. Meyer, editors, *Evolutionary Robotics (Proc. EvoRob'98)*, volume 1468 of *LNCS*, pages 71–86, Berlin, 1998. Springer.

[66] Takashi Ikegami and Takashi Hashimoto. Active mutation in self-reproducing networks of machines and tapes. *Artificial Life*, 2(3):305–318, 1995.

[67] Takashi Ikegami and Takashi Hashimoto. Replication and diversity in machine-tape coevolutionary systems. In Christopher G. Langton and Katsunori Shimohara, editors, *Artificial Life V*, pages 426–433, Cambridge, MA, 1997. MIT Press.

[68] P. Inverardi and A.L. Wolf Formal specification and analysis of software architectures using the chemical abstract machine model. *IEEE Transactions on Software Engineering*, 21(4):373–386, 1995.

[69] S. Jain and S. Krishna. Autocatalytic sets and the growth of complexity in an evolutionary model. *Phys. Rev. Lett.*, 81(25):5684–5687, 1998.

[70] Yasui Kanada. Combinatorial problem solving using randomized dynamic tunneling on a production system. In *1995 IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century*, volume 4, pages 3784–9, New York, NY, 1995. IEEE.

[71] Yasui Kanada and Masao Hirokawa. Stochastic problem solving by local computation based on self-organization paradigm. In *27th Hawaii International Conference on System Science*, pages 82–91, 1994.

[72] S. A. Kauffman Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, 22:437–467, 1969.

[73] S. A. Kauffman Autocatalytic sets of proteins. *J. Theor. Biol.*, 119:1–24, 1986.

[74] S. A. Kauffman *The Origins of Order: Self-Organization and Selection in Evolution.* Oxford University Press, New York, 1993.

[75] S. A. Kauffman and W. Macready. Search strategies for applied molecular evolution. *J. Theor. Biol.*, 173(4):427–440, 1995.

[76] Stuart A. Kauffman and Sonke Johnson. Co-evolution to the edge of chaos: Coupled fitness landscapes, poised states, and co-evolutionary avalanches. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 325–370, Redwood City, CA, 1992. Addison-Wesley.

[77] John R. Koza Artificial life: Spontaneous emergence of self-replicating and evolutionary self-improving computer programs. In C.G. Langton, editor, *Artificial Life III*, pages 225–262, Reading, MA, 1994. Addison-Wesley.

[78] R. Laing. Artificial organisms and autonomous cell rules. *Journal of Cybernetics*, 2(1):38–49, 1972.

[79] R. Laing. The capabilities of some species of artificial organism. *Journal of Cybernetics*, 3(2):16–25, 1973.

[80] R. Laing. Some alternative reproductive strategies in artificial molecular machines. *J. Theor. Biol.*, 54:63–84, 1975.

[81] R. Laing. Automaton introspection. *J. of Computer and System Sciences*, 13(2):172–83, 1976.

[82] R. Laing. Automaton models of reproduction by self-inspection. *J. Theor. Biol.*, 66:437–56, 1977.

[83] C. G. Langton Self-reproduction in cellular automata. *Physica D*, 10D(1-2):135–44, 1984.

[84] Stanislaw Lem. *Summa technologiae.* Wydawnictwo Literackie, Kraków, 1964.

[85] R. J. Lipton. Speeding up computation via molecular biology. Technical report, Princton University, 1995.

[86] J. D. Lohn, S.P. Colombano, J. Scargle, D. Stassinopoulos, and G. L. Haith. Evolution of catalytic reaction sets using genetic algorithms. In *Proc. IEEE International Conference on Evolutionary Computation*, pages 487–492, New York, NY, 1998. IEEE.

[87] Marek W. Lugowski Computational metabolism: Towards biological geometries for computing. In Christopher G. Langton, editor, *Artificial Life*, pages 341–368, Redwood City, CA, 1989. Addison-Wesley.

[88] C. D. Mao, T. H. LaBean, J. H. Reif, and N. C. Seeman. Logical computation using algorithmic self-assembly of dna triple-crossover molecules. *Nature*, 407(6803):493–496, 2000.

[89] Bern Mayer and Steen Rasmussen. Self-reproduction of dynamical hierachies in chemical systems. In C. Adami, R. Belew, H. Kitano, and C. Taylor, editors, *Artificial Life VI*, pages 123–129, Cambridge, MA, 1998. MIT Press.

[90] Bernd Mayer and Steen Rasmussen. The lattice molecular automaton (LMA): A simulation system for constructive molecular dynamics. *Int. J. of Modern Physics C*, 9(1):157–177, 1998.

[91] J. S. McCaskill Polymer chemistry on tape: A computational model for emergent genetics. Internal report, MPI for Biophysical Chemistry, Göttingen, Germany, 1988.

[92] J. S. McCaskill, H. Chorongiewski, D. Mekelburg, U. Tangen, and U. Gemm Configurable computer hardware to simulate long-time self-organization of biopolymers. *Ber. Bunsenges. Phys. Chem.*, 98(9):1114–1114, 1994.

[93] Barry McMullin and Francisco J. Varela Rediscovering computational autopoiesis. In P. Husbands and I. Harvey, editors, *Fourth European Conference on Artificial Life*, pages 38–47, Cambridge, MA, 1997. MIT Press.

[94] Bernd Meyer, Gottfried Köhler, and Steen Rasmussen. Simulation and dynamics of entropy-driven, molecular self-assembly process. *Phys. Rev. E*, 55(4):4489–4500, 1997.

[95] Harold C. Morris Typogenetics: A logic for artificial life. In Christopher G. Langton, editor, *Artificial Life*, pages 341–368. Addison-Wesley, 1989.

[96] Robert B. Nachbar Molecular evolution: Automated manipulation of hierarchical chemical topology and its application to average molecular structures. *Genetic Programming and Evolvable Machines*, 1(1/2):57–94, 2000.

[97] N. Ono and T. Ikegami. Self-maintenance and self-reproduction in an abstract cell model. *J. Theor. Biol.*, 206(2):243–253, 2000.

[98] A. N. Pargellis The evolution of self-replicating computer organisms. *Physica D*, 98(1):111–127, 1996.

[99] A. N. Pargellis The spontaneous generation of digital "life". *Physica D*, 91(1-2):86–96, 1996.

[100] Gheorghe Păun. Computing with membranes. *J. of Computer and System Sciences*, 61(1):108–143, 2000.

[101] Alan S. Perelson and Stuart A. Kauffman *Molecular Evolution on Rugged Landscapes: Proteins, RNA and the Immune System*, volume 9. Addison-Wesley, Reading, MA, USA, 1991.

[102] Steen Rasmussen, Carsten Knudsen, and Rasmus Feldberg. Dynamics of programmable matter. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 211–291, Redwood City, CA, February 1992. Addison-Wesley.

[103] Steen Rasmussen, Carsten Knudsen, Rasmus Feldberg, and Morten Hindsholm. The coreworld: Emergence and evolution of cooperative structures in a computational chemistry. *Physica D*, 42:111–134, 1990.

[104] Steen Rasmussen and J. Smith Lattice polymer automaton. *Ber. Bunsenges. Phys. Chem.*, 98(9):1185–1193, 1994.

[105] Hilmar Rauhe. Investigations of the origin of self-replicating digital organisms. Diploma thesis, Max-Planck-Institut für Züchtungsforschung, Köln, 1995. (in german).

[106] Thomas S. Ray. Is it alive or is it GA. In Richard K. Belew and Lashon B. Booker, editors, *Proc. Fourth International Conference on Genetic Algorithms (ICGA'91)*, pages 527–534, San Mateo, CA, USA, 13-16 July 1991. Morgan Kaufmann.

[107] Thomas S. Ray An approach to the synthesis of life. In Christopher G. Langton, Charls Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 371–408, Redwood City, CA, 1992. Addison-Wesley.

[108] M. Rizki and M. Conrad. Computing the theory of evolution. *Physica D*, 22:83–99, 1986.

[109] A. Sali, E. Shakhnovich, and M. Karplus How does a protein fold? *Nature*, 369(6477):248–251, 1994.

[110] A. Sali, E. Shakhnovich, and M. Karplus Kinetics of protein folding. a lattice model study of the requirements for folding to the native state. *J. Mol. Biol.*, 235(5):1614–1636, 1994.

[111] H. Sayama A new structurally dissolvable self-reproducing loop evolving in a simple cellular automata. *Artificial Life*, 5(4):343–65, 2000.

[112] P. Schuster and K. Sigmund. Replicator dynamics. *J. Theor. Biol.*, 100:533–8, 1983.

[113] D. Segre, D. Ben-Eli, D. W. Deamer, and D. Lancet. The lipid world. *Orig. Life Evol. Biosph.*, 31(1-2):119–145, 2001.

[114] D. Segré, D. Lancet, O. Kedem, and Y. Pilpel Graded autocatalysis replication domain (GARD): Kinetic analysis of self-replication in mutually catalytic sets. *Orig. Life Evol. Biosph.*, 28(4-6):501–514, 1998.

[115] M. A. Shackleton and C. S. Winter. A computational architecture based on cellular processing. In Mike Holcombe and Ray Paton, editors, *Information Processing in Cells and Tissues*, pages 261–272, New York, NY, 1998. Plenum Press.

[116] Nicholas D. Socci and Jose Nelson Onuchic. Folding kinetics of proteinlike heteropolymers. *Journal of Chemical Physics*, 101(2):1519–1528, 1995.

[117] Pietro Speroni di Fenizio. A less abstract artficial chemistry. In M. A. Bedau, J. S. McCaskill, N. H. Packard, and Steen Rasmussen, editors, *Artificial Life VII*, pages 49–53, Cambridge, MA, 2000. MIT Press.

[118] Peter F. Stadler, Walter Fontana, and John H. Miller. Random catalytic reaction networks. *Physica D*, 63:378–392, 1993.

[119] Y. Suzuki, J. Takabayashi, and H. Tanaka. Investigation of an ecological system by using an abstract rewriting system on multisets. In *to appear*, 1999.

[120] Yasuhiro Suzuki and Hiroshi Tanaka. Symbolic chemical system based on abstract rewriting and its behavior pattern. *Artificial Life and Robotics*, 1:211–219, 1997.

[121] Yasuhiro Suzuki and Hiroshi Tanaka. Order parameter for a symbolic chemical system. In C. Adami, R. Belew, H. Kitano, and C. Taylor, editors, *Artificial Life VI*, pages 130–139, Cambridge, MA, 1998. MIT Press.

[122] T. Szuba. Evaluation measures for the collective intelligence of closed social structures. In *International Conference on Intelligent Systems and Semiotics (ISAS'97)*, pages 401–406, Gaithersburg, MD, USA, 1997.

[123] T. Szuba. A molecular quasi-random model of computations applied to evaluate collective intelligence. *Futur. Gener. Comp. Syst.*, 14(5-6):321–339, 1998.

[124] T. Szuba and A. Stras. Evaluation of the inference power of a closed social structure with the help of the random prolog processor. In *Fifth International Conference on the Practical Application of Prolog*, pages 369–389, 1997.

[125] T. Szuba and R. Straš Parallel evolutionary computing with the random prolog processor. *J. Parallel Distrib. Comput.*, 47(1):78–85, 1997.

[126] Ch. Tanford The hydrophobic effect and the organization of living matter. *Science*, 200(4345):1012, 1978.

[127] U. Tangen, L. Schulte, and J. S. McCaskill A parallel hardware evolvable computer polyp. In K. L. Pocek and J. Arnold, editors, *IEEE Symposium on FPGAs for Custopm Computing Machines*, Los Alamitos, CA, 1997. IEEE Computer Society.

[128] P. D. Taylor and L. Jonker Evolutionary stable strategies and game dynamics. *Math. Biosci.*, 40:154–56, 1978.

[129] Marcel Thürk. *Ein Modell zur Selbstorganisation von Automatenalgorithmen zum Studium molekularer Evolution.* PhD thesis, Universität Jena, naturwissenschaftliche Fakultät, 1993.

[130] Carlo Vanderzande. *Lattice Models of Polymers.* Cambridge University Press, Cambridge, UK, 1998.

[131] F. J. Varela, H. R. Maturana, and R. Uribe Autopoiesis: The organization of living systems. *BioSystems*, 5(4):187–196, 1974.

[132] L. Varetto Typogenetics: An artificial genetic system. *J. Theor. Biol.*, 160(2):185–205, 1993.

[133] L. Varetto Studying artificial life with a molecular automaton. *J. Theor. Biol.*, 193(2):257–285, 1998.

[134] G. M. Whitesides, J. P. Mathias, and C. T. Seto. Molecular self-assembly and nanochemistry: A chemical strategy for the synthesis of nanostructures. *Science*, 254:1312–1319, 1991.

[135] T. Yamamoto and K. Kaneko. Tile automaton: A model for an architecture of a living system. *Artif. Life*, 5(1):37–76, 1999.

[136] K.-P. Zauner and M. Conrad. Conformation-driven computing: Simulating the context-conformation-action loop. *Supramolecular Science*, 5(5-6):791–794, 1998.

[137] Klaus-Peter Zauner and Michael Conrad. Simulating the interplay of structure, kinetics, and dynamics in complex biochemical networks. In R. Hofestädt, T. Lengauer, M. Löffler, and D. Schomburg, editors, *Proc. German Conference on Bioinformatics (GCB'96)*, pages 336–338, Leipzig, 1996. Universität Leipzig.

[138] Milan Zeleny. Self-organization of living systems: A formal model of autopoiesis. *International Journal of General Science*, 4:13–28, 1977.

[139] Jens Ziegler, Peter Dittrich, and Wolfgang Banzhaf. Towards a metabolic robot controller. In Mike Holcombe and Ray Paton, editors, *Information Processing in Cells and Tissues*, pages 305–318, New York, NY, 1998. Plenum Press.

[140] J. Ziegler and W. Banzhaf. Evolving a "nose" for a robot. In *Workshop Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*. Morgan Kaufmann, 2000.